

AD-A273 056



Mississippi State
UNIVERSITY

2

Center for Air Sea Technology

DESIGN DOCUMENT
AND DATABASE SPECIFICATIONS
for the

NAVAL INTERACTIVE DATA ANALYSIS SYSTEM

(NIDAS) Version 1.0

Technical Note 01-94
29 October 1993

S DTIC
ELECTE
NOV 10 1993
A

Prepared for: Naval Oceanographic Office (Code DOST)
Stennis Space Center, Mississippi 39529

Contract Number: NAS13-564, Order No. 11

Approved for public release; distribution is unlimited.
Mississippi State University Center for Air Sea Technology
Stennis Space Center, MS 39529-6000



93-27506

DESIGN DOCUMENT
and DATABASE SPECIFICATION
FOR THE
NAVAL INTERACTIVE DATA ANALYSIS SYSTEM
(NIDAS)

NIDAS Version 1.0 (29 October 1993)

CONTRACT NO: NAS13-564 D.O. 11

CONTRACT DELIVERABLE: 1b/1c

Prepared for:

NAVAL OCEANOGRAPHIC OFFICE
CODE DOST
STENNIS SPACE CENTER, MS 39529

Prepared by:

Mississippi State University
Center for Air Sea Technology
Building 1103, Room 233
Stennis Space Center, MS 39529-6000

Acknowledgements

Special Note of Thanks: The Center for Air Sea Technology gratefully acknowledges the special attention, clear guidance and encouragement of Mr. Steve Haeger, Naval Oceanographic Office. Without Mr. Haeger's assistance, application design and development would have experienced delays at every turn.

UNIX is a trademark of American Telephone and Telegraph (AT&T), Incorporated

SUN, SUNOS and SparcStation are trademarks of Sun Microsystems, Incorporated

Motif is a trademark of the Open Software Foundation

X-Windows is a trademark of the Massachusetts Institute of Technology

Empress is a trademark of Empress Software, Incorporated

ag/X Toolmaster is a trademark of UNIRAS, Incorporated

Contributors to the design of the NIDAS application are:

Mr. Dharmesh Krishnamagaru, Principle Investigator/Software Engineer
Mr. Vishnu Mohan Das, Software Engineer
Mr. Ramesh Krishnamagaru, Senior Software Engineer
Mr. Valentine Anantharaj, Associate Scientist

Contributors to the NIDAS Software Design Document are:

Mr. Dharmesh Krishnamagaru, Software Engineer/Project Manager
Mr. Vishnu Mohan Das, Software Engineer
Mr. Micheal S. Foster, Editor
Ms. LeAna Dusang, Structure, Content and Graphical Artwork

DTIC QUALITY INSPECTED 8

Accession For	
NTIS	CRA&I <input checked="" type="checkbox"/>
DTIC	TAB <input type="checkbox"/>
Unannounced <input type="checkbox"/>	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Table of Contents

	Page Number
Cover Sheet	i
Acknowledgements	ii
Table of Contents	iii
List of Figures	vi
1 SCOPE	1
1.1 Identification	1
1.2 System Overview	1
1.3 Document Overview	1
2 REFERENCED DOCUMENTS	2
3 PRELIMINARY DESIGN	2
3.1 CSCI Overview	2
3.1.1 CSCI Architecture	3
3.1.1.1 CSC-1	3
3.1.1.2 CSC-2	4
3.1.1.3 CSC-3	4
3.1.1.4 CSC-4	4
3.1.1.5 GUI-DRM (CSC-1/CSC-2) Internal Interface	5
3.1.1.6 GUI-DIM (CSC-1/CSC-3) Internal Interface	5
3.1.1.7 GUI-NSM (CSC-1/CSC-4) Internal Interface	5
3.1.1.8 NSM-DIM (CSC-4/CSC-3) Internal Interface	5
3.1.2 System States and Modes	5
3.1.3 Memory and Processing Time Allocation	5
3.2 CSCI Design Description	5
3.2.1 Graphical User Interface (GUI) (CSC-1)	6
3.2.2 Data Retrieval Module (DRM) (CSC-2)	6
3.2.3 Data Interactive Module (DIM) (CSC-3)	7
3.2.4 NIDAS Session Module (NSM) (CSC-4)	8
4 DETAILED DESIGN	9
4.1 CSC-1 - The NIDAS Graphical User Interface (GUI)	9
4.1.1 NIDAS Main Window	9
4.1.1.1 NIDAS Main Window Design Specification/Constraints	10
4.1.1.2 NIDAS Main Window Design	10
4.2 CSC-2 - The NIDAS Data Retrieval Module (DRM)	11
4.2.1 The NIDAS Data Interface	11
4.2.1.1 NIDAS DATA Interface Design Specification/Constraints	11
4.2.1.2 NIDAS DATA Interface Design	11
4.2.1.2.1 Bathymetry Data Selection	11
4.2.1.2.2 Bathymetry Display	12
4.2.1.2.3 Bathymetry Options	12
4.2.1.2.4 MODAS Data Selection	12

4.2.1.2.5	MODAS Data Display	13
4.2.1.2.6	MODAS Data Options	13
4.2.1.2.7	Climatology Data Selection	13
4.2.1.2.8	Climatology Data Display	14
4.2.1.2.9	Climatology Data Options	14
4.2.1.2.10	MOODS Data Selection	14
4.2.1.2.11	MOODS Data Display	15
4.2.1.2.12	MOODS Data Options	16
4.2.1.2.13	GOODS Data Selection	16
4.2.1.2.14	GOODS Data Display	16
4.2.1.2.15	GOODS Data Options	17
4.2.1.2.16	Fronts Data Selection	17
4.2.1.2.17	Fronts Data Display	18
4.2.1.2.18	Fronts Data Options	18
4.2.1.2.19	Eddy Data Selection	18
4.2.1.2.20	Eddy Data Display	18
4.2.1.2.21	Eddy Data Options	18
4.2.1.2.22	MODAS Profile Data Selection	19
4.2.1.2.23	MODAS Profile Data Display	19
4.2.1.2.24	MODAS Profile Data Options	19
4.2.1.2.25	Climatology Profile Data Selection	19
4.2.1.2.26	Climatology Profile Data Display	19
4.2.1.2.27	Climatology Profile Data Option	20
4.2.1.2.28	Image Data Selection	20
4.2.1.2.29	Image Data Display	20
4.2.1.2.30	Image Data Options	21
4.2.1.2.31	NIDAS-3D Data Selection	21
4.2.1.2.32	NIDAS-3D Data Display	21
4.2.1.2.33	NIDAS-3D Data Options	22
4.3	CSC-3	22
4.3.1	The NIDAS Interactive Interface	22
4.3.1.1	NIDAS Interactive Interface Design Specifications/Constraints	22
4.3.1.2	NIDAS Interactive Interface Design	22
4.3.1.2.1	Polygon	22
4.3.1.2.2	Zoom	23
4.3.1.2.3	Transect	23
4.3.1.2.5	SingleMODAS	25
4.4.1	The NIDAS Session Interface	25
4.4.1.1	NIDAS Session Interface Design Specifications/Constraints	25
4.4.1.2	NIDAS Session Interface Design	25
5	NIDAS DATA	26
5.1	The NIDAS Database	26
5.2	NIDAS User Configuration Data	26
6	NIDAS DATA FILES	27
7	REQUIREMENTS TRACEABILITY	27
8	NOTES	28

APPENDICES

A	Glossary of Terms	A-1
B	List of Acronyms	B-1
C	NIDAS Database Specification	C-1
D	Functional and Design Requirements for the NIDAS Relational Database Management System (RDBMS)	D-1
E	NIDAS Development Environment	E-1
F	Modifications and Changes to the NIDAS Design	F-1
G	NIDAS Structures	G-1
H	User Default File	H-1

LIST OF FIGURES:

Figure 1. Top-level modular structure of the Naval Interactive Data Analysis System (NIDAS)	3
Figure 2. Illustration of the NIDAS Graphical User Interface (GUI) (CSC-1) display screen	6
Figure 3. Illustration of the Data Retrieval Module (DRM) display window	7
Figure 4. The Data Interactive Module (DIM) Data Editing options window	8
Figure 5. NIDAS Session Module (NSM) "Synthetic Admin Info" pop-up window	8
Figure 6. NIDAS detail design and connectivity	9
Figure 7. The NIDAS Main Window	11
Figure 8. DRM "Options" pop-up for Bathymetry, MODAS, Image, Climatology and NIDAS-3D data	12
Figure 9. DRM "Data" pop-up window for the MODAS data type	13
Figure 10. DRM "Data" pop-up window for Climatology data types	14
Figure 11. DRM "Data" pop-up window for the MOODS data type	15
Figure 12. DRM "Options" pop-up window for the MOODS and GOODS data types	16
Figure 13. DRM "Data" pop-up for the GOODS data type	17
Figure 14. DRM "Data" pop-up window for Front and Eddy data types	17
Figure 15. DRM "Options" pop-up window for Front and Eddy data types	18
Figure 16. DRM "Options" pop-up window for the MODAS Profile data type	19
Figure 17. DRM "Options" pop-up window for the Climatology Profile data type	20
Figure 18. DRM "Data" pop-up window for the Image data type	20
Figure 19. The DRM "Data" pop-up window for the NIDAS-3D data type	21
Figure 20. The DIM "Transect" pop-up window	24
Figure 21. The DIM "Synthetic Profile" pop-up window	24
Figure 22. The DIM plotting and editing options pop-up window	24
Figure 23. The NSM "New Session" pop-up window	26
Figure 24. NSM "Unclosed Session" pop-up window	26

NIDAS SOFTWARE DESIGN DOCUMENT

1 SCOPE

1.1 Identification

Computer Software Configuration Item (CSCI): Naval Interactive Data Analysis System (NIDAS)

Version: 1.0

Release Date: To Be Determined at a Later Date

Contract No: NAS13-564 D.O. 11

Contractor: Mississippi State University
Center for Air Sea Technology
J. H. Corbin, Director
Building 1103, Room 233
Stennis Space Center, MS 39529-6000
Telephone: (601) 688-2561
Facsimile: (601) 688-7100

Principle Investigator: Mr. Dharmesh Krishnamagaru, Mississippi State University
Center for Air Sea Technology, Stennis Space Center, MS
39529-6000. (MSU Proposal No. 93-3-467)
Telephone: (601) 688-7141
Facsimile: (601) 688-7100

1.2 System Overview

NIDAS provides interactive blending and overlaying capabilities for a broad range of environmental data types and formats. The final product output of NIDAS is three-dimensional gridded fields of temperature and salinity, constructed from 1) data derived by the user internally within NIDAS and 2) data obtained from external sources (numerical models, observations and climatological databases).

1.3 Document Overview

The purpose of this document is to define and describe the structural framework and logical design of the software components/units which will be integrated into the major computer software configuration item (CSCI) identified as NIDAS, Version 1.0. The preliminary design is based on functional specifications and requirements identified in the 26 January 1993 Statement of Work prepared by the Naval Oceanographic Office (NAVOCEANO) and distributed as a request for proposal by the National Aeronautics and Space Administration on 17 March 1993. The contents and format of this document is specified by Department of Defense (DOD)-STD 2167A.

Appendix A contains a glossary of terms used within this document. Appendix B is a listing of acronyms used within this document. The NIDAS relational database specification is contained in Appendix C. Appendix D contains functional and design requirements for the NIDAS relational database management system (RDBMS). Refer to Appendix E for a description of the

NIDAS development environment. Modifications and changes to the NIDAS Preliminary Design are found in Appendix F. Appendix G contains listings of all relevant source code structures. Appendix H provides information about the NIDAS configuration file.

2 REFERENCED DOCUMENTS

Note: The section(s) or subsection(s) of this document wherein a reference is cited appears as parenthetical information following each document listed in this section.

2.1. DOD-STD 2167A "Defense System Software Development", AMSC No. N4327, 29 Feb 88. (1.3)

2.2. National Aeronautics and Space Administration letter DA20 of 17 Mar 92, Subject: Contract NAS13-564; (Mississippi Research Consortium); Request for Proposal for the Naval Interactive Data Analysis System (NIDAS). (1.3)

2.3. Naval Oceanographic Office Statement of Work of 26 Jan 93, Task Title: Development of the Naval Interactive Data Analysis System (NIDAS). (1.3)

2.4. Mississippi State University proposal number 93-3-467 of 25 Mar 93, Title: Development of the Naval Interactive Data Analysis System (NIDAS), MSU Office of Sponsored Programs, P.O. Box 6156, Mississippi State, Mississippi 39762. (1.1)

3 PRELIMINARY DESIGN

3.1 CSCI Overview

The Naval Interactive Data Analysis System (NIDAS), a stand-alone system, is the major Computer Software Configuration Item (CSCI) to be developed by this project. Functional requirements, as identified by the sponsor, were not defined in the context of a modular approach to software development. Therefore, the subordinate tasks necessary to fulfill these requirements have been divided among/assigned to the internal Computer Software Components (CSC) for accomplishment. The top-level (simplistic) NIDAS architecture is illustrated in Figure 1.

There are four principle CSC's within the NIDAS structure:

- a. **Graphical User Interface (GUI)** - incorporates window management, user interface and display functionality;
- b. **Data Retrieval Module (DRM)** - provides functional data management using relational RDBMS technology;
- c. **Data Interactive Module (DIM)** - incorporates data processing, application of interactive methods and algorithms to the data, and graphical (visualization) processing of the data.
- d. **NIDAS Session Module (NSM)** - establishes a session for creating and maintaining oceanographic provinces and associated parameter profile(s).

NIDAS has two external interfaces:

- a. The **User-GUI** interface supports 1) user control of NIDAS using interactive techniques and 2) response/feedback to the user in the form of data display (graphical or numerical) and status indicators.
- b. The **Files-GUI** interface provides access to files and data that are not resident within the internal NIDAS RDBMS.

3.1.1 CSCI Architecture

Figure 1 illustrates the NIDAS top-level (simplistic) module and external interface architecture.

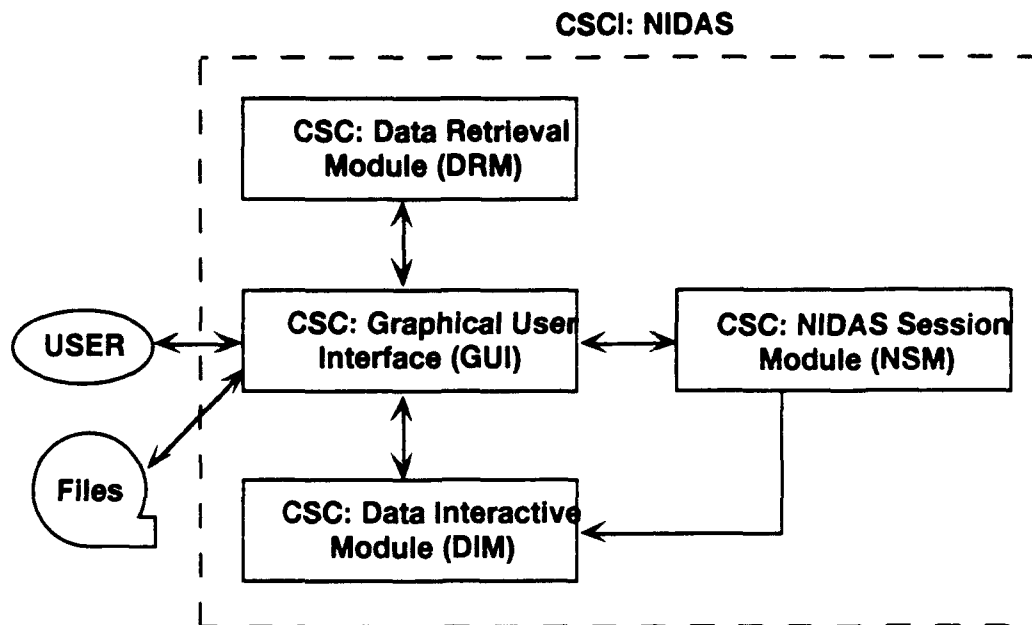


Figure 1. Top-level modular structure of the Naval Interactive Data Analysis System (NIDAS). NIDAS is the Computer Software Configuration Item (CSCI). There are four Computer Software Components (CSC).

3.1.1.1 CSC-1

Graphical User Interface (GUI). The NIDAS GUI supports and manages the link between the user and NIDAS. Through the GUI, the user exercises all available NIDAS control options. NIDAS provides displays to the user for interpretation and interactive response. The GUI integrates non-developmental software as follows:

a. **X-Windows** (proprietary): Manages all controls and display windows. The X-Windows client/server model supports remote user access to NIDAS using network protocol via the UNIX operating system.

b. **Motif Widget Toolkit** (proprietary): Provides a widely accepted standard inventory of window components (widgets) that is pleasing to the eye and easy to interpret.

Functionally, the GUI:

- Exercises direct, centralized control over the DRM, DIM, and NSM;
- Monitors activities of the DRM, DIM, and NSM via their external interfaces;
- Serves as the agent for internal DRM/DIM/NSM communications;
- Intercepts, interprets and routes user interactive commands;
- Provides status information and feedback to the user;

- f. Provides a windowing environment for visualization of data, display of control elements and intercepting user interactive commands.
- g. Receives and interprets user input via the keyboard or mouse pointing device.

3.1.1.2 CSC-2

Data Retrieval Module (DRM). The NIDAS DRM is primarily tasked with managing access to and communications with the internal NIDAS RDBMS. In addition, the DRM prepares simple data displays that conform to user-selectable options and passes them to the GUI for presentation in a window. When a data display is designated for interactive manipulation, it is passed to the DIM via the GUI. The DRM receives data management instructions from the GUI (user). In turn, through the Naval Environmental Operational Nowcast system (NEONS), the DRM translates these instructions into Structured Query Language (SQL) commands to the RDBMS. The DRM is also responsible for allocating the internal memory space for data retrieved from the database or data destined for database ingestion. The DRM employs the following non-developmental software in accomplishing its tasks:

- a. **UNIRAS ag/X Toolmaster** (proprietary): Handles graphical representation and visualization of data displays;
- b. **Empress RDBMS** (proprietary): Performs low-level management of NIDAS' RDBMS;
- c. **Naval Environmental Operational Nowcast System (NEONS)** (government provided software): Performs high level management of the NIDAS RDBMS via embedded SQL commands to the underlying Empress RDBMS. NEONS also embodies the data model used by NIDAS' RDBMS.

3.1.1.3 CSC-3

Data Interactive Module (DIM). The DIM is responsible for modifying and manipulating the data in response to user interactive commands. This module produces the final 3-D product fields, which may be retained in memory or stored in files. These memory-resident products and files may subsequently be ingested into the RDBMS via the DRM. In addition, the DIM supplies the GUI with real-time, updated data displays that reflect user interaction. As in the DRM, graphical representation and visualization within the DIM is accomplished using **UNIRAS ag/X Toolmaster** (proprietary non-developmental software).

3.1.1.4 CSC-4

NIDAS Session Module (NSM). The NSM is primarily responsible for establishing a new work session or selecting an existing closed/unclosed session. The session must be established in order to insert oceanographic provinces and their associated profiles. As with other forms of data, sessions are stored in the database. The user cannot insert a province and its defining polygon to a closed session. The NSM supplies the DIM with the sessions required to perform synthetic 3-dimensional data field construction within a given region. The NSM employs the following non-developmental software in accomplishing its tasks:

- a. **UNIRAS ag/X Toolmaster** (proprietary): Handles graphical representation and visualization of data displays;
- b. **Empress RDBMS** (proprietary): Performs low-level management of NIDAS' RDBMS;

c. **Naval Environment Operational Nowcast System (NEONS)** (government provided software): Performs high level management of NIDAS' RDBMS via embedded SQL commands to the underlying Empress RDBMS. NEONS also embodies the data model used by NIDAS' RDBMS.

3.1.1.5 GUI-DRM (CSC-1/CSC-2) Internal Interface

For design purposes, the GUI-DRM internal interface consists of all facilities for transfer of control or data between the GUI and the DRM. The GUI-DRM interface is an abstract construct employed to group those actions which pass information between the two modules.

3.1.1.6 GUI-DIM (CSC-1/CSC-3) Internal Interface

For design purposes, the GUI-DIM internal interface consists of all facilities for transfer of control or data between the GUI and the DIM. The GUI-DIM internal interface embodies the same functionality and abstract character as the GUI-DRM interface.

3.1.1.7 GUI-NSM (CSC-1/CSC-4) Internal Interface

For design purposes, the GUI-NSM internal interface consists of all facilities for transfer of control or data between the GUI and the NSM. The GUI-NSM internal interface embodies the same functionality and abstract character as the GUI-DRM interface.

3.1.1.8 NSM-DIM (CSC-4/CSC-3) Internal Interface

For design purposes, the NSM-DIM internal interface consists of all facilities for communication and transfer of control or data between the DIM and the NSM. The NSM-DIM interface is an abstract construct employed to group those actions which pass information between the two modules.

3.1.2 System States and Modes

NIDAS is an interactive software application. The application is always in the event-driven state. As with all event-driven software applications, NIDAS may assume either of two execution states (modes): 1) processing mode and 2) rest (idle) mode. The NIDAS default state is the rest mode. When not processing data in response to user input, NIDAS automatically reverts to the rest mode and awaits the next user command or input.

3.1.3 Memory and Processing Time Allocation

The interactive, event-driven nature of NIDAS precludes a quantitative description of memory allocation and processing time among NIDAS CSC's. It is therefore considered sufficient to state that NIDAS responds to user demands by allocating memory, swap space and processor time within the limitations imposed by available memory and UNIX operating system constraints.

3.2 CSCI Design Description

The CSCI (NIDAS) consists of four CSC modules with functionality as described above. Functional requirements for the CSCI, as stated by the sponsor, cannot always be accomplished wholly and completely within a single CSC. For instance, the final NIDAS product (a three-dimensional gridded field of temperature and salinity) requires the cooperative contribution of all

four NIDAS CSC modules. The remainder of this section identifies NIDAS requirements by CSC and discusses the software design employed to achieve the required functionality. The CSCI incorporates all NIDAS functional requirements (NFR) and design requirements (NDR) as presented in Section 7 of this document. In addition, the CSCI achieves specific design requirements which are not accomplished by any CSC. These are NDR1, NDR2, NDR3, NDR4 and NDR10.

3.2.1 Graphical User Interface (GUI) (CSC-1)

CSC-1 is responsible for providing the visual display link between the user and the NIDAS main interactive display. The preliminary design for the NIDAS main interactive display window is illustrated in Figure 2. While other CSC modules manage their own suite of window displays, the GUI is the ultimate destination of the functionality achieved through user interaction with them. The DRM (CSC-2), DIM (CSC-3) and NSM (CSC-4) perform the preparatory data selection, data processing, and data management steps required prior to the appearance of visual displays within the GUI. The GUI and its sub-level components achieve the following specific functional and design requirements, either wholly or in concert with other CSC's (see Section 7): NFR1, NFR2, NFR3, NFR24, NFR26, NFR27, NFR28, NFR29, NFR30, NFR31, NFR32, NFR33, NFR34, NFR35, NFR36, NFR37, NFR38, NFR39, NFR40, NFR41, NFR42, NDR9 and NDR11.

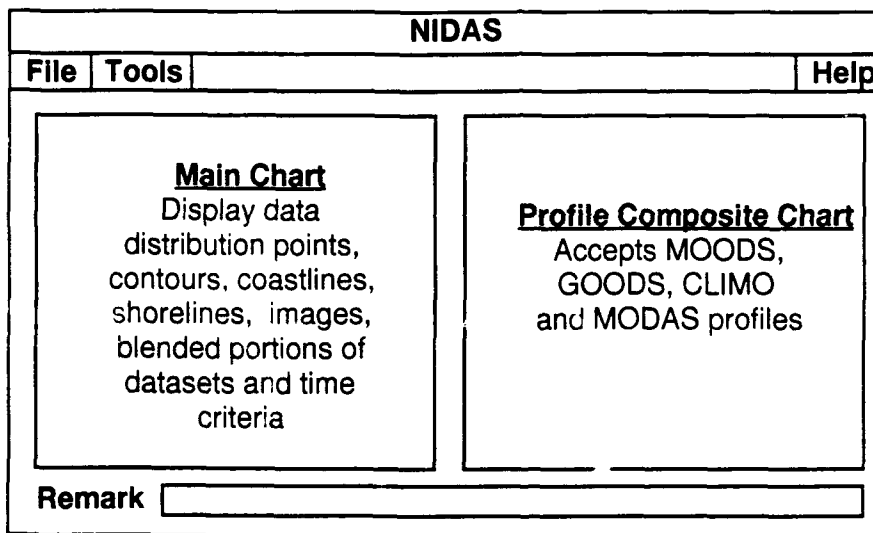


Figure 2. Illustration of the NIDAS Graphical User Interface (GUI) (CSC-1) display screen. The GUI is the final destination for all interactive coordination between the user and the user-controllable processes resident in the Data Retrieval Module (DRM) (CSC-2), Data Interactive Module (DIM) (CSC-3), and the NIDAS Session Module (NSM) (CSC-4).

3.2.2 Data Retrieval Module (DRM) (CSC-2)

The Data Retrieval Module (DRM) (CSC-2) controls the selection and display of data for the NIDAS CSCI. User interaction with the DRM is handled via the Data Selection pop-up window shown in Figure 3, below. There are three choices for each selectable data type listed in

the Data Selection pop-up window, "Display", "Data", and "Options". The "Display" buttons select data types to be viewed in the "Main Chart" portion of the GUI window. The "Data" buttons produce dialog pop-up windows, tailored to each data type, that allow selection of available parameters contained in the dataset. The "Options" buttons produce data dialog pop-up windows, tailored to each data type, offering other user options such as color. The DRM and its sub-level components achieve the following specific functional and design requirements, either wholly or in concert with other CSC's (see Section 7): NFR1, NFR3, NFR4, NFR7, NFR8, NFR9, NFR10, NFR11, NFR12, NFR13, NFR14, NFR15, NFR16, NFR17, NFR18, NFR19, NFR20, NFR21, NFR22, NFR25, NFR26, NFR38, NDR5, NDR6, NDR8, NDR11, NDR5, NDR6, NDR8 and NDR11.

Data Selection			
	Display	Data	Options
BATHY	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MODAS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CLIMO	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
COAST	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MOODS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
GOODS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
FRONTS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
EDDYS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
MODASPROF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CLIMOPROF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
IMAGE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
NIDAS3D	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 3. Illustration of the Data Retrieval Module (DRM) display window. This pop-up window lists the available data types and provides corresponding Display, Options and Data buttons.

3.2.3 Data Interactive Module (DIM) (CSC-3)

The DIM provides capability for manipulating and editing selected data. Options are provided to support (1) identification of data subsets by constructing polygon boundaries, (2) zooming of data profiles for improved interpretation/analysis, (3) creation and display of a transection, or slice, through the dataset and (4) creation of synthetic datasets(subsets). These choices are made within the pop-up display window shown in Figure 4. The DIM and its sub-level components achieve the following specific functional and design requirements, either wholly or in concert with other CSC's (see Section 7): NFR1, NFR3, NFR5, NFR6, NFR7, NFR22, NFR24, NFR27, NFR29, NFR30, NFR31, NFR32, NFR33, NFR34, NFR35, NFR36, NFR37, NFR38, NFR41, NDR9 and NDR11.

Data Editing	
<input checked="" type="checkbox"/>	POLYGON
<input checked="" type="checkbox"/>	ZOOM
<input checked="" type="checkbox"/>	TRANSECT
<input checked="" type="checkbox"/>	SYNTHETIC
<input checked="" type="checkbox"/>	SINGLE MODAS

Figure 4. The Data Interactive Module (DIM) Data Editing options window.

3.2.4 NIDAS Session Module (NSM) (CSC-4)

The NSM allows the user to establish a new NIDAS session or to access an unclosed session. A NIDAS session provides the facilities for construction of provinces and their associated profiles. Provinces (and associated profiles) may be selected from previously closed sessions. Unclosed sessions may be displayed for construction of provinces. High level control of the NSM is achieved by means of options available through the NSM pop-up display window shown in Figure 5. The NSM and its sublevel components achieve the following NIDAS functional and design requirements, either wholly or in concert with other CSC's (see Section 7): NFR3, NFR5, NFR6, NFR7, NFR24, NFR30, NFR31, NFR32, NFR33, NFR34, NFR35, NFR44, NFR45, NFR46, NFR47.

Synthetic Admin Info	
Initiate	
<input checked="" type="checkbox"/> New	
<input checked="" type="checkbox"/> Unclosed	
Session Boundary Overlay	
<input type="checkbox"/> Display	<input type="checkbox"/> Data <input type="checkbox"/> Options
Repeat	
<input checked="" type="checkbox"/> Repeat	
Terminate	
<input checked="" type="checkbox"/> Close	
<input checked="" type="checkbox"/> TurnOff	
<input type="button" value="Reset"/>	

Figure 5. NIDAS Session Module (NSM) "Synthetic Admin Info" pop-up window.

4 DETAILED DESIGN

Figure 6 illustrates the functional connectivity within NIDAS and forms the basis for the detailed design of the NIDAS CSCI. The preliminary design (see Section 3) provided a broad overview of the framework and functional design of the NIDAS CSCI and its four major CSC's [Graphical User Interface (GUI), Data Retrieval Module (DRM), Data Interactive Module (DIM) and NIDAS Session Module (NSM)]. In this section, each CSC will be described in detail.

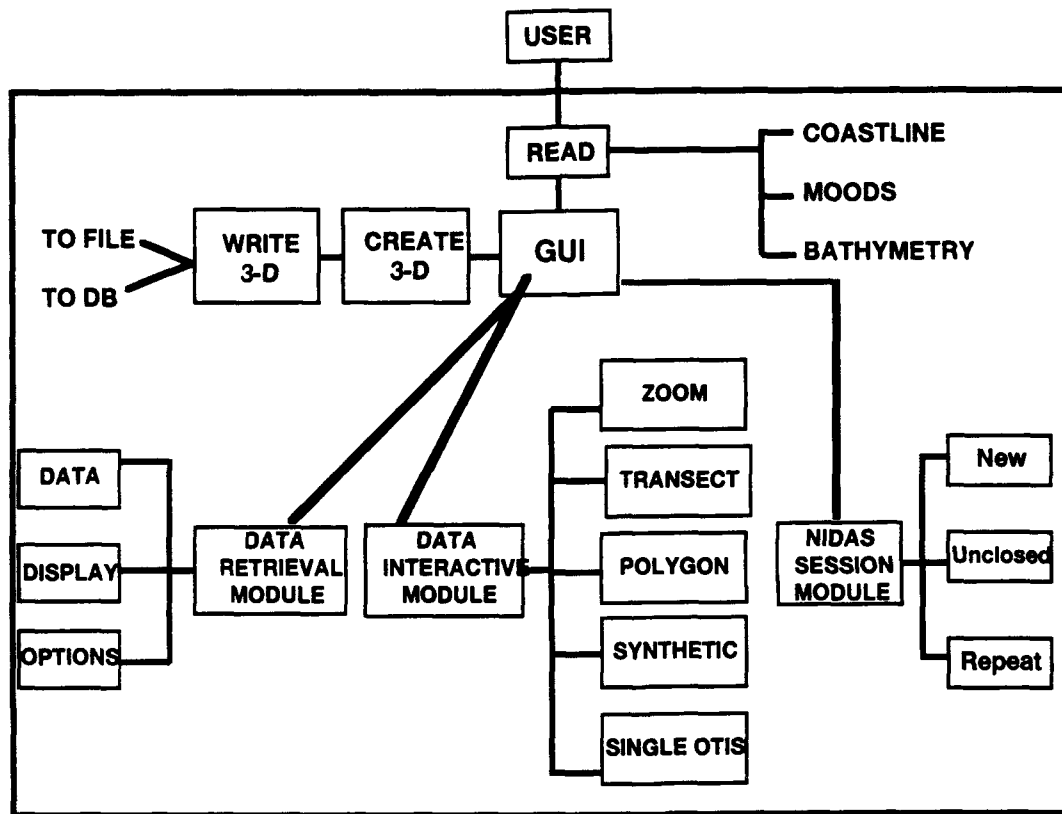


Figure 6. NIDAS detail design and connectivity.

4.1 CSC-1 - The NIDAS Graphical User Interface (GUI)

4.1.1 NIDAS Main Window

The design of the Main Window CSU is subdivided into five areas: 1) the title area; 2) the "Main Chart" which provides a geographical display; 3) the "Profile Chart" which displays MOODS, GOODS, CLIMO, and MODAS profiles; 4) a pull-down menu bar containing various options for interacting with the "Main Chart" and "Profile Chart" windows, either separately or simultaneously; and 5) a "Remark" textbox which communicates important messages, such as errors to the user.

4.1.1.1 NIDAS Main Window Design Specification/Constraints

The design criteria for the NIDAS Main Window CSU are listed in Section 7, "Requirements Traceability". This CSU is constrained by the following requirements: 1) the X-server must be opened and available for display; 2) the database must be opened and be available for reading; and 3) UNIRAS ag/X Toolmaster must be active for handling graphics functionality.

4.1.1.2 NIDAS Main Window Design

The NIDAS Main Window design employs the X, Motif, NEONS, and UNIRAS ag/X Toolmaster libraries. Input to the NIDAS Main Window is via the NIDAS structure. The function **allocMemory()** allocates memory for the NIDAS structure. The function **initNidasStruct()** initializes NIDAS structure data elements. The function **readDefaultValues()** retrieves user-defined default values for various NIDAS structure elements from the file **nidas_config.def**. Bathymetry, coastline and MOODS datasets are retrieved from the database. The function **readBathymetryData()** retrieves bathymetry data and stores it in the NIDAS Bathy structure. The **retrieveMoodsData()** function retrieves MOODS data and stores it in the NIDAS Moods structure. The function **createCoastlinePixmap()** is called to draw the proper coastline on the "Main Chart" Window display.

The function **frontPageLayout()** creates the layout of the NIDAS Main Window as shown in Figure 7. The function **dataForm()** creates the layout for DRM functionality as shown in Figure 3. The function **editForm()** creates the layout for DIM functionality as shown in Figure 4. The function **syntheticForm()** creates the layout for the NSM pop-up window as shown in Figure 5. The button "Create 3D" under the "Tools" pull-down menu calls the function **createNidasModasGrid()** which creates the 3D grid and stores it into the database. The button "Write 3D Grid" calls the function **setup3DGridOption()** which writes NIDAS-3D grid or newly constructed MODAS grid to a file. The button "Export" calls the function **setupExportData()** which supports export of the data defined in a polygon region. The Main Window reports via the "Remark" textbox when it cannot find the default values file or if one of the necessary libraries. The repaint button removes all the polygons drawn on both window. The NIDAS Main Window is not dynamically configured to the size of the monitor screen. If the monitor screen measures less than fourteen inches diagonally, portions of the Main window may be clipped by the screen border.

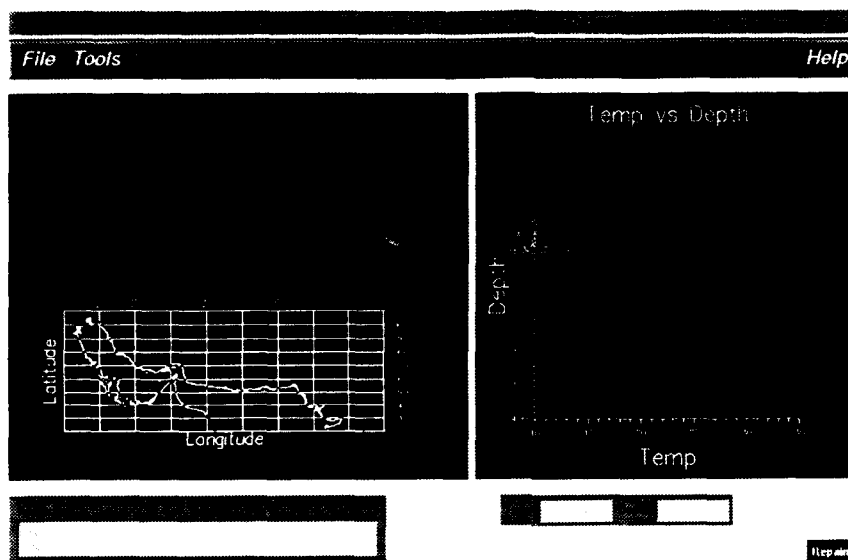


Figure 7. The NIDAS Main Window illustrating the "Main Chart" (left) and "Profile Chart" (right) displays.

4.2 CSC-2 - The NIDAS Data Retrieval Module (DRM)

4.2.1 The NIDAS Data Interface

The purpose of the NIDAS Data Interface is to provide data retrieval and display and data manipulation options for the following data types: bathymetry, coastlines, MODAS, climatology, MOODS, GOODS, fronts, eddies, MODAS profiles, climatology profiles, satellite images, and NIDAS 3-dimensional grid fields.

4.2.1.1 NIDAS DATA Interface Design Specification/Constraints

The design requirements for the NIDAS Data Interface may be found in Section 7. Constraints on this interface are the same as for the NIDAS Main Window CSU.

4.2.1.2 NIDAS DATA Interface Design

The NIDAS Data Interface is written in the C programming language. It is linked at compile time to the X, Motif, NEONS, and ag X Toolmaster libraries. Before accessing this interface, memory for the NIDAS data structure must be allocated and the structure must be initialized.

4.2.1.2.1 Bathymetry Data Selection

Bathymetry data is automatically read from the database and stored in the NIDAS Bathy structure in accordance with parameters contained in the NIDAS default configuration file. There is no other opportunity for a user to control the selection of bathymetric data.

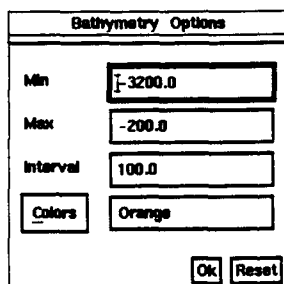
4.2.1.2.2 Bathymetry Display

When the Bathymetry "Data" button is activated, the function **displayBathymetry()** is executed. This function allocates memory to a pixmap in which bathymetry contours are drawn by calling the function **createBathymetryPixmap()** as described in the subparagraph below. The pixmap is copied to the Main Window's "Main Chart" for display of bathymetry contours. When the Bathymetry "Data" button is deactivated, the pixmap containing bathymetry contours is removed from the "Main Chart" and destroyed.

The function **createBathymetryPixmap()** first determines the size of the NIDAS region and allocates memory for a new pixmap. **createBathymetryPixmap()** extracts bathymetry data from the original bathymetry array for the region. It then sets the scale, data range, the color and the labels; then, it draws the bathymetry contours using UNIRAS ag/X Toolmaster library graphics routines. Finally, **createBathymetryPixmap()** releases the memory allocated for the new array.

4.2.1.2.3 Bathymetry Options

When the Bathymetry "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. This function creates and displays the "Bathymetry Options" pop-up window as illustrated in Figure 8. The function **bathyOtisGdemOptions()** is also executed by the MODAS, Image, Climatology and NIDAS-3D "Options" buttons. The Bathymetry "Options" button is automatically deactivated when the "Ok" button is clicked to terminate and destroys the pop-up window. This functional feature also applies to the MODAS, Image, Climatology and NIDAS-3D "Options" buttons. The "Bathymetry Options" pop-up window allows setting of minimum ("Min"), maximum ("Max"), contour interval ("Interval") and contour color ("Colors").



Bathymetry Options	
Min	-3200.0
Max	-200.0
Interval	100.0
Colors	Orange
<div>Ok Reset</div>	

Figure 8. DRM "Options" pop-up for Bathymetry, MODAS, Image, Climatology and NIDAS-3D data types.

4.2.1.2.4 MODAS Data Selection

When the MODAS "Data" button is activated, the function **setupOtisData()** is executed which, in turn, calls the function **otisGdemNidas3dLayout()**. **otisGdemNidas3dLayout()** creates and displays the "MODAS Data" pop-up window shown in Figure 9. The "MODAS Data" pop-up window provides textbox listings for Levels, Parameters and Dates for MODAS. Default parameters are listed. One entry from each textbox list must be selected in order to retrieve a data set from the database. The MODAS "Data" button is automatically deactivated when the "Exit" button in the "MODAS Data" pop-up window is clicked.

OtisGdemNidas3dLayout() executes the function **readOtisVol()** to read the dataset and store it in the NIDAS MODAS structure. The function **readOtisVol()** employs NEONS volumetric and grid functions to accomplish its data retrieval chores.

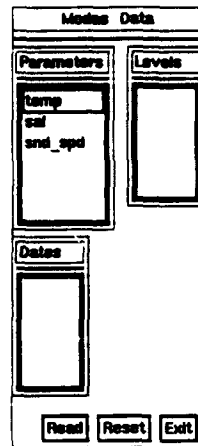


Figure 9. DRM "Data" pop-up window for the MODAS data type.

4.2.1.2.5 MODAS Data Display

When the MODAS "Display" button is activated, the function **displayOtis()** is executed. **displayOtis()** allocates memory for a pixmap where MODAS contours are to be drawn. Next, **displayOtis()** calls the function **createOtisPixmap()** to draw the MODAS contours on the pixmap as described in the next paragraph. **displayOtis()** then copies the pixmap to the Main Window's "Main Chart" where the contour pixmap is displayed. When the MODAS "Display" button is deactivated, the MODAS contour pixmap is removed from the "Main Chart" and destroyed.

The function **createOtisPixmap()** determines the size of the NIDAS region and allocates memory for a new pixmap array of that size. The function extracts the data from the existing array for the region and stores it in the new array. **createOtisPixmap()** sets the scale, data range, interval, contour color and the labels. Finally, it draws the contours using UNIRAS ag/X Toolmaster graphics library routines and releases the memory allocated for the new array.

4.2.1.2.6 MODAS Data Options

When the MODAS "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. The purpose and methodology of **bathyOtisGdemOptions()** is described in paragraph 4.2.1.2.3 (Bathymetry Options), above.

4.2.1.2.7 Climatology Data Selection

When the Climatology "Data" button is activated, the function **setupGdemData()** which, in turn, calls the function **otisGdemNidas3dLayout()** is executed. The purpose and methodology of **otisGdemNidas3dLayout()** is described in paragraph 4.2.1.2.4 (MODAS Data Selection), above. The associated "Climo Data" pop-up window is illustrated in Figure 10.

Once the Climatology level, parameter and season have been defined, **setupGdemData()** executes the function **gdemRd()** to read the Climatology dataset and store it in the NIDAS MODAS structure. The function **gdemRd()** employs NEONS volumetric and grid functions to accomplish its data retrieval chores.

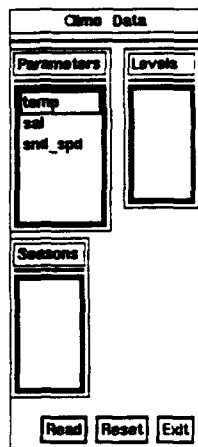


Figure 10. DRM "Data" pop-up window for Climatology data types.

4.2.1.2.8 Climatology Data Display

When the Climatology "Display" button is activated, the function **displayGdem()** is executed. The function **displayGdem()** allocates memory for a pixmap where Climatology contours may be drawn. The function **displayGdem()** then calls the function **createGdemPixmap()** which draws the Climatology contours within the pixmap as described in the next subparagraph. The function **displayGdem()** then copies the pixmap to the "Main Chart" where the contour plot is displayed. When the Climatology "Display" button is deactivated, the Climatology contour pixmap is removed from the Main Window and destroyed.

The function **createGdemPixmap()** determines the size of the NIDAS region, allocates memory for a new array of the size of the NIDAS region, moves data from the existing regional array to the new array, sets the scale, data range, color and the labels, and draws the contours using UNIRAS ag/X Toolmaster graphics library routines. Finally, **createGdemPixmap()** releases the memory allocated for the new array.

4.2.1.2.9 Climatology Data Options

When the Climatology "Options" button is activated, the function **bathyOtistGdemOptions()** is executed. The purpose and methodology of **bathyOtistGdemOptions()** is described in paragraph 4.2.1.2.3 (Bathymetry Options), above and illustrated in Figure 8.

4.2.1.2.10 MOODS Data Selection

When the MOODS "Data" button is activated, the function **createMoodsDataLayout()** is executed. The function **createMoodsDataLayout()** produces and displays a data selection window for the MOODS data type as shown in Figure 11. Initially, this window displays default element values obtained from the NIDAS MOODS structure. The default element values can be

changed prior to retrieving MOODS data from the database (which is initiated by clicking the "Ok" button in the lower right corner of the data selection window).

MOODS data is read from the database by the function **retrieveMoodsData()**. This function retrieves the data using LLT routines from the NEONS library. The function **storeMoodsVal()** allocates memory and stores the data in the NIDAS MOODS structure. The MOODS "Data" button is automatically deactivated when the data is retrieved and stored.

Data Selection		Selection Status	
Classification 1010010 Unclassified; unlimite 1100010 Unclassified 10001000 delete class	Instrument Type 2 Mechanical Bathytherm 5 new 10 Some unknown electron 11 Expendable bathytherm 12 Air deployed expendable	<input type="checkbox"/> Minimum <input type="checkbox"/> Maximum	
Source Code 2 FNOC message data 4 NODC SDII - through 198 6 FNOC XBT in SPOT frame 7 NODC STD dat high resol 8 British - MBT and Xbt da	Months January February March April May	Latitude	22.00 31.00
		Longitude	47.00 74.00
		Time	1/1/1900:0 10/22/1993:9
		Classification	10001000 10001000
		Month	1 2
		Parameters	3 3
		Cruise Id	0 151375
		Instrument Type	0 33
		Source Code	2 31
		Water Depth	0 3950
Cruise Id: <input type="text"/> 1 Water Depth: <input type="text"/> Latitude: <input type="text"/> Longitude: <input type="text"/> <input type="checkbox"/> Time Number Of Parameters: <input type="text"/>		<input type="button" value="Read"/> <input type="button" value="Reset"/> <input type="button" value="Exit"/>	

Figure 11. DRM "Data" pop-up window for the MOODS data type.

4.2.1.2.11 MOODS Data Display

When the MOODS "Display" button is activated, the function **displayMoods()** is executed. The function **displayMoods()** allocates memory for two pixmaps, one for plotting MOODS profile locations and one for plotting the actual profiles. The function **displayMoods()** uses the function **createMoodsLocationPixmap()** to plot profile locations and the function **createMoodsProfilesPixmap()** to plot the profile. The location pixmap is then copied to the Main Window "Main Chart" display and the profile plot pixmap is copied to the "Profile Composite Chart" display. When the MOODS "Display" button is deactivated, location plots for MOODS data are removed from the "Main Chart", MOODS profiles are removed from the "Profile Composite Chart" and, finally, the two pixmaps are destroyed.

The function **createMoodsLocationPixmap()** establishes the proper scale and calls the function **plotMoodsLatLon()** to plots MOODS profile locations. The function **createMoodsProfilesPixmap()** establishes the proper scale and employs the function **drawMoodsProfile()** to plot the actual MOODS profiles.

4.2.1.2.12 MOODS Data Options

When the MOODS "Options" button is activated, the function `createMoodsGoodsOptions()` is executed. The function `createMoodsGoodsOptions()` displays a pop-up window, as illustrated in Figure 12, for setting available options for both MOODS and GOODS datasets. When MOODS (or GOODS) option selection is complete, the MOODS (or GOODS) "Options" button is automatically deactivated.

Figure 12. DRM "Options" pop-up window for the MOODS and GOODS data types.

4.2.1.2.13 GOODS Data Selection

When the GOODS "Data" button is activated, the function `createGoodsDataLayout()` is executed. The function `createGoodsDataLayout()` displays a pop-up window. Figure 13, for use in setting GOODS dataset selection criteria. Initially, the displayed default values for selection elements are from the NIDAS GOODS structure. The pop-up window allows modification of these default element values which are used internally to identify and retrieve the desired GOODS dataset.

GOODS data is retrieved from the database by the function `retrieveGoodsData()` using LLT routines from the NEONS library. The function `storeGoodsVal()` allocates memory for the GOODS dataset and places the address of the data in the NIDAS GOODS structure. The GOODS "Data" button is automatically deactivated after GOODS data has been retrieved and stored.

4.2.1.2.14 GOODS Data Display

When the GOODS "Display" button is activated, the function `displayGoods()` is executed. The function `displayGoods()` allocates memory for two pixmaps, one for plotting GOODS profile locations and one for plotting the actual profiles. The function `displayGoods()` uses the function `createGoodsLocationPixmap()` to plot profile locations and the function `createGoodsProfilesPixmap()` to plot the actual profiles. The location pixmap is then copied to the Main Window "Main Chart" display and the profile plot pixmap is copied to the "Profile Composite Chart" display. When the GOODS "Display" button is deactivated, location plots for GOODS data are removed from the "Main Chart", GOODS profiles are removed from the "Profile Composite Chart" and, finally, the two pixmaps are destroyed.

The function **createGoodsLocationPixmap()** establishes the proper scale and calls the function **plotGoodsLatLon()** to plots GOODS profile locations. The function **createGoodsProfilesPixmap()** establishes the proper scale and employs the function **drawGoodsProfile()** to plot the actual GOODS profiles.

Figure 13. DRM "Data" pop-up for the GOODS data type.

4.2.1.2.15 GOODS Data Options

When the GOODS "Options" button is activated, the function **createMoodsGoodsOptions()** is executed. The function **createMoodsGoodsOptions()** displays a pop-up window, as illustrated in Figure 12, for setting available options for both GOODS and MOODS datasets. When GOODS (or MOODS) option selection is complete, the GOODS (or MOODS) "Options" button is automatically deactivated.

4.2.1.2.16 Fronts Data Selection

When the Fronts "Data" button is activated, the function **freddyStructLayout()** is executed. The function **freddyStructLayout()** creates and displays the "Fronts Data" pop-up window for Fronts and Eddies shown in Figure 14. The "Fronts Data" pop-up window allows selection of a date from a textbox listing of dates for available datasets. Date selection identifies the proper Front dataset to be retrieved when the "Ok" button is clicked. The Fronts "Data" button is automatically deactivated when the "Ok" button in the "Fronts Data" pop-up window is clicked.

Figure 14. DRM "Data" pop-up window for Front and Eddy data types.

4.2.1.2.17 Fronts Data Display

When the Fronts "Display" button is activated, the function **displayFronts()** is executed. The function **displayFronts()** allocates memory for a fronts pixmap in which fronts are drawn. The function **displayFronts()** calls the function **createFrontsPixmap()** to draw the fronts on the pixmap. The function **displayFronts()** then copies the pixmap to the "Main Chart" for display. When the Fronts "Display" button is deactivated, the fronts pixmap is removed from the "Main Chart" and destroyed.

The function **createFrontsPixmap()** sets the scale on the "Main Chart" and calls the function **getFrontData()** which uses Line routines from the NEONS library to retrieve the data from the database. Finally, **getFrontData()** plots the fronts dataset on the pixmap.

4.2.1.2.18 Fronts Data Options

When the Fronts "Options" button is activated, the function **freddyStructLayout()** is executed. **freddyStructLayout()** creates and displays the "Fronts Options" pop-up window shown in Figure 15. The only option allowed is selection of the color to be used by plotting routines. The Fronts "Options" button is automatically deactivated when the pop-up window's "Ok" button is clicked.

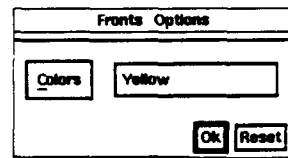


Figure 15. DRM "Options" pop-up window for Front and Eddy data types.

4.2.1.2.19 Eddy Data Selection

When the Eddy "Data" button is activated, the function **freddyStructLayout()** is executed. **freddyStructLayout()** creates the "Fronts Data" pop-up window shown in Figure 12. The "Fronts Data" pop-up window allows selection of a date from a textbox listing of dates for available datasets. Date selection identifies the proper Eddy dataset to be retrieved when the "Ok" button is clicked. The Eddy "Data" button is automatically deactivated when the "Ok" button from the "Fronts Data" pop-up window is clicked.

4.2.1.2.20 Eddy Data Display

When the Eddy "Display" button is activated, the function **displayEddy()** is executed. The function **displayEddy()** allocates memory for a pixmap in which eddies are drawn. The function **displayEddy()** calls the function **createEddyPixmap()** to draw the eddies on the pixmap. The pixmap is copied to the "Main Chart" for display. When the Eddy "Display" button is deactivated, the eddy pixmap is removed from the "Main Chart" display and destroyed.

The function **createEddyPixmap()** sets the scale for the "Main Chart" display and calls the function **getEddyData()** which uses Line functions from the NEONS library to retrieve the data from the database. Finally, **getEddyData()** draws the eddies on the pixmap.

4.2.1.2.21 Eddy Data Options

When the Eddy "Options" button is activated, the function **freddyStructLayout()** is executed. The function **freddyStructLayout()** creates and displays the "Fronts Options" pop-up window shown in Figure 15. The only option allowed is selection of the color to be used by plotting routines. The Eddy "Options" button is automatically deactivated when the pop-up window's "Ok" button is clicked.

4.2.1.2.22 MODAS Profile Data Selection

The MODAS Profile "Data" button is an inactive (null) button. No action occurs when this button is clicked. It is provided as a contingency for future NIDAS design modifications.

4.2.1.2.23 MODAS Profile Data Display

When the MODAS Profile "Display" button is activated, the function **displayOtisProfile()** is called to set a flag indicating the desire to display MODAS Profile data when a polygon is constructed by routines in the DIM. No plotting occurs in response to activation of this button. **displayOtisProfile()** sends a flag-setting message which is handled by the DIM.

4.2.1.2.24 MODAS Profile Data Options

When the MODAS Profile "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. This function creates and displays the pop-up window illustrated in Figure 16. The function **bathyOtisGdemOptions()** is also executed by the Bathy, Image, and Climatology "Options" buttons. The MODAS Profile "Options" button is automatically deactivated when the "Ok" button is clicked to terminate and destroy the "Bathymetry Options" pop-up window. Only color is selected by the user.

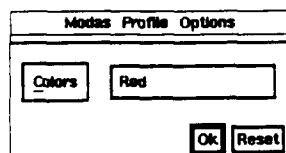


Figure 16. DRM "Options" pop-up window for the MODAS Profile data type.

4.2.1.2.25 Climatology Profile Data Selection

The Climatology Profile "Data" button is an inactive (null) button. No action occurs when this button is clicked. It is provided as a contingency for future NIDAS design modifications.

4.2.1.2.26 Climatology Profile Data Display

When the Climatology Profile "Display" button is activated, the function **displayGdemProfile()** is called to set a flag indicating the desire to display Climatology Profile data when a polygon is constructed by routines in the DIM. No plotting occurs in response to activation of this button. The function **displayGdemProfile()** sends a flag-setting message which is handled by the DIM.

4.2.1.2.27 Climatology Profile Data Option

When the Climatology Profile "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. The function **bathyOtisGdemOptions()** creates and displays the pop-up window illustrated in Figure 17. The function **bathyOtisGdemOptions()** is also executed by the Bathy, Image, and MODAS Profile "Options" buttons. The Climatology Profile "Options" button is automatically deactivated when the "Ok" button is clicked to terminate and destroy the "Bathymetry Options" pop-up window. Only color is selected by the user.

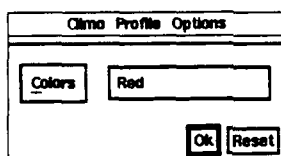


Figure 17. DRM "Options" pop-up window for the Climatology Profile data type.

4.2.1.2.28 Image Data Selection

When the Image "Data" button is activated, the function **imageDataLayout()** is executed. The function **imageDataLayout()** creates and displays the "Image Data" pop-up window as shown in Figure 18. The "Image Data" pop-up window allows selection of a date which is keyed to an image dataset within the database. When the "Ok" button within the "Image Data" pop-up window is clicked, the pop-up window is destroyed and the Image "Data" button is automatically deactivated.

Image datasets are read from the database and stored in the NIDAS Image structure by the function **readImage()**, which uses the image routines found in the NEONS library.

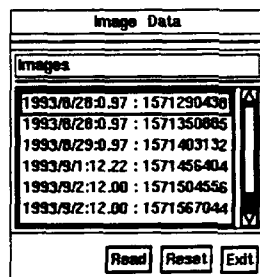


Figure 18. DRM "Data" pop-up window for the Image data type.

4.2.1.2.29 Image Data Display

When the Image "Display" button is activated, the function **displayImage()** is executed. The function **displayImage()** allocates memory to a pixmap where an image can be drawn. **displayImage()** calls the function **createImagePixmap()** transfers the image onto the pixmap and copies it to the "Main Chart" for display. When the Image "Display" button is deactivated, the image pixmap is removed from the "Main Chart" and the pixmap is destroyed.

The function **createImagePixmap()** first determines the size of NIDAS region and allocates memory to a new pixmap array based on the size of the region. The function

createImagePixmap() then extracts the data from the original array for the region and stores it in the new array. The function **createImagePixmap()** sets the scale and color, and draws the image using functions from the UNIRAS ag/X Toolmaster graphics library. Finally, **createImagePixmap()** releases the memory allocated for the new array.

4.2.1.2.30 Image Data Options

When the Image "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. The function **bathyOtisGdemOptions()** creates and displays the "Bathymetry Options" pop-up window illustrated in Figure 8. The function **bathyOtisGdemOptions()** is also executed by the Bathy, MODAS Profile and Climatology Profile "Options" buttons. The Image "Options" button is automatically deactivated when the "Ok" button is clicked to terminate and destroy the "Bathymetry Options" pop-up window. This functional feature also applies to the Bathymetry, MODAS Profile and Climatology Profile "Options" buttons. The "Bathymetry Options" pop-up window allows setting of minimum ("Min"), maximum ("Max"), interval ("Interval").

4.2.1.2.31 NIDAS-3D Data Selection

When the NIDAS-3D "Data" button is activated, the function **setupNIDAS3dData()** is executed. The function **setupNIDAS3dData()** calls the function **otisGemNidas-3dLayout()** which produces the data selection pop-up window illustrated in Figure 19.

The function **readNidas3dVol()** retrieves NIDAS-3D data from the database and stores it in the NIDAS-3D structure. The function **readNidas3dVol()** employs the Volumetric and Grid functions contained in the NEONS library.

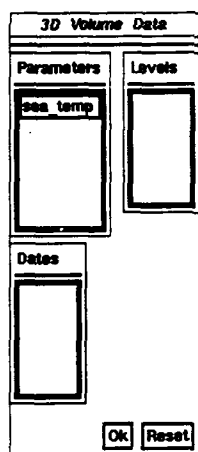


Figure 19. The DRM "Data" pop-up window for the NIDAS-3D data type.

4.2.1.2.32 NIDAS-3D Data Display

When the NIDAS-3D "Display" button is activated, the function **displayNidas3d()** is executed. The function **displayNidas3d()** allocates memory for a pixmap within which NIDAS-3D contours may be drawn. The function **displayNidas3d()** calls the function **createNidas3dPixmap()** to perform the NIDAS-3D contour drawing chores.

displayNidas3d() then copies the pixmap to the "Main Chart" for display. When the NIDAS-3D "Display" button is deactivated, the NIDAS-3D contour pixmap is removed from the "Main Chart" and destroyed.

The function **createNidas3dPixmap()** determines the size of the NIDAS region and allocates memory for a new array based on region size. It then extracts the data from the original array and stores it in the new array. The function **createNidas3dPixmap()** sets the scale, data range, the color and the labels for the NIDAS-3D pixmap, and draws the contours using the UNIRAS ag/X Toolmaster graphics library. Finally, **createNidas3dPixmap()** releases the memory allocated for the new array.

4.2.1.2.33 NIDAS-3D Data Options

When the NIDAS-3D "Options" button is activated, the function **bathyOtisGdemOptions()** is executed. The function **bathyOtisGdemOptions()** creates the "3D Volume Data" pop-up window illustrated in Figure 8. The NIDAS-3D "Options" button is automatically deactivated when the "Ok" button within the "Bathymetry Options" (Figure 8) pop-up window is pressed.

4.3 CSC-3 - The NIDAS Data Interactive Module (DIM)

4.3.1 The NIDAS Interactive Interface

The purpose of the NIDAS Interactive Interface is to support user interaction with the data. Five interactive services are provided: 1) Zooming, 2) drawing polygons, 3) building transections, and 4) constructing synthetic profiles and 5) drawing Single MODAS locations.

4.3.1.1 NIDAS Interactive Interface Design Specifications/Constraints

The design requirements for the NIDAS Interactive Interface are listed in Section 7. Constraints applicable to this CSC are the same as those for the GUI CSC.

4.3.1.2 NIDAS Interactive Interface Design

The NIDAS Interactive Interface is coded in the C programming language. It is linked with the X, Motif, NEONS and ag/X Toolmaster libraries. Currently, the NIDAS data structure must be initialized before using the NIDAS Interactive Interface.

4.3.1.2.1 Polygon

When the "Polygon" button is activated, the function **polygonEdit()** is executed. The function **polygonEdit()** attaches the callback function **mainDrawingPress()** to the "Main Chart" and the function **profileDrawingPress()** to the "Profile Chart". When the "Polygon" button is deactivated, the callback functions **mainDrawingPress()** and **profileDrawingPress()** are removed from the "Main Chart" and "Profile Chart", respectively.

The function **mainDrawingPress()** uses the leftmost mouse button to highlight the border color of the "Main Chart". The middle mouse button is used to specify the locations of polygon corners. The rightmost mouse button is used to close the polygon and call the function

drawFlaggedLocations() which plots the locations of data profiles lying inside the polygon in the "Main Chart" and plots the corresponding profiles in the "Profile Chart".

The function **profileDrawingPress()** uses the leftmost mouse button to highlight the border color of the "Profile Chart". The middle mouse button is used to specify the locations of polygon corner. The rightmost button is used to close the polygon and call the function **drawFlaggedLocations()** which plots those profiles lying within the polygon on the "Profile Chart" and their corresponding locations on the "Main Chart".

4.3.1.2.2 Zoom

When the "Zoom" button is activated, the function **zoomEdit()** is executed. **zoomEdit()** attaches the callback functions **zoomDrawingPress()**, **zoomDrawingMotion()**, and **zoomDrawingRelease()** to the "Main Chart". When the "Zoom" button is deactivated, **zoomEdit()** detaches these callback functions from the "Main Chart".

The function **zoomDrawingPress()** draws an initial rectangle when the leftmost mouse button is pressed. The function **zoomDrawingMotion()** erases the old rectangle and draws a new rectangle as long as: 1) there is mouse motion, and 2) the leftmost mouse button is depressed. The function **zoomDrawingRelease()**: 1) draws the final rectangle when the leftmost mouse button is released, 2) computes the new zoom region, and 3) calls the function **zoomGraphics()** to draw the newly scaled graphics on the "Main Chart" and "Profile Chart" for the zoomed region.

4.3.1.2.3 Transect

When the "Transect" button is activated, the function **transectEdit()** is executed. **transectEdit()** attaches the callback function **transectDrawingPress()** to the "Main Chart". When the "Transect" button is deactivated, the function **transectEdit()** detaches **transectDrawingPress()** from the "Main Chart".

The function **transectDrawingPress()** acquires the start- and end-points of the transect from the cursor locations identified by two successive clicks of the leftmost mouse button in the "Main Chart". The function **transectDrawingPress()** calls the function **createTransectOptions()**, which creates and displays the pop-up window shown in Figure 20. This pop-up window provides options for drawing transect contours and profiles. The function **createTransectWindow()** creates two windows, one for drawing transect contours and one for plotting transect profiles.

4.3.1.2.4 Synthetic

When the "Synthetic" button is activated, the function **syntheticProfile()** is executed. **syntheticProfile()** creates and displays the "Synthetic Profile" pop-up window shown in Figure 21. When the "Repeat" option is selected, the function **drawBookkeepingChart()** is executed. The function **drawBookkeepingChart()** creates a "repeat session" window and draws all the provinces for a repeat session. Upon selection of a province from the "repeat session" window, the function **repeatProfileLayout()** displays the pop-up window shown in Figure 22, which provides five options as follows:

1) the "Plot Province" button executes the function **redrawProvince()**, which redraws the selected province on the "Main Chart";

2) the "Plot Synthetic" button calls the function **plotPreviousSynthetic()** which plots the synthetic profile on the "Profile Chart";

3) the "Edit Synthetic Profile" button accesses the function **pickDrawingPress()** and **editDrawingPress()** for editing synthetic profiles using procedures identical to those used for editing real profiles;

4) the "Draw New Synthetic" button executes the function **newSyntheticDrawingPress()** which allows construction of a new synthetic profile in the "Profile Chart";

5) the "Scale" button accesses the function **axisProfileOptions()** via the function **scaleProfileWindow()** to allow delineation of maximum and minimum values for the x- and y-axis.

When the "Last Polygon" option is selected from the "Synthetic Profile" pop-up window, it attaches the callback function **syntheticDrawingPress()** to the "Profile Chart" which permits construction of a profile for each of the polygons drawn.

Figure 20. The DIM "Transect" pop-up window.

Figure 21. The DIM "Synthetic Profile" pop-up window.

Figure 22. The DIM plotting and editing options pop-up window.

4.3.1.2.5 SingleMODAS

When the "SingleMODAS" button is activated, the function **singleOtisEdit()** is executed. The function **singleOtisEdit()** attaches the callback function **mainOtisLocGetPress()** to the "Main Chart". When the "SingleMODAS" button is deactivated, it calls the function **SingleOtisEdit()** to remove the callback function **mainOtisLocGetPress()** from the "Main Chart". The function **mainOtisLocGetPress()** obtains the location of the profile inside a polygon on the "Main Chart" when the leftmost mouse button is clicked, and calls the function **drawSingleOtisLocation()** to draw the profiles in the "Profile Chart".

4.4 CSC-4 - The NIDAS Session Module(NSM)

4.4.1 The NIDAS Session Interface

The purpose of the NIDAS Session Interface (NSM) is to establish a session to which provinces and their associated synthetic profiles can be linked.

4.4.1.1 NIDAS Session Interface Design Specifications/Constraints

The design requirements for the NIDAS Session interface are listed in Section 7. The constraints binding this CSC are the same as those for the GUI CSC.

4.4.1.2 NIDAS Session Interface Design

The Session Interface is coded in C programming language. It is linked with the X, NEONS, and ag/X Toolmaster libraries at compile time. The NIDAS synthetic structure is associated with the NSM and must be initialized prior to NSM activation. The function **syntheticForm()** creates a pop-up window for the NSM as shown in Figure 5. When the "New" button is clicked, a new session is established by calling the function **setupNewSessionInfo()**. This function creates the pop-up window shown in Figure 23, wherein the user must enter the data necessary for new session initiation. To retrieve an old session, the "Unclosed" button is clicked, executing the function **setupSessionInfo()**, which provides a pop-up window that lists unclosed sessions as shown in Figure 24. Similarly, the "Data" button under "Session Boundary Overlay" calls the function **setupSessionInfo()**. The "Repeat" button also calls **setupSessionInfo()**; but this time, a list of closed and unclosed sessions is provided for selection. The user must select a new or unclosed session to work with synthetic profiles within the DIM CSC. An unclosed or new session can be closed by clicking the "Close" button. The "Turnoff" button prevents entry of new provinces and profiles to an unclosed or new session.

New Session

Session Region

Type	Mode	Classification
<input type="checkbox"/> Production <input type="checkbox"/> Cllm_Const <input type="checkbox"/> Demo <input type="checkbox"/> Request <input type="checkbox"/> Other	<input type="checkbox"/> Practice <input type="checkbox"/> Real	<input type="checkbox"/> Classified <input type="checkbox"/> Unclassified

ModelType	Voild	Date	StampTime
MODAS	: 172553591	: 1993/5/31:0.00	: 10/18/1993 08:03:16

Ok Edit

Figure 23. The NSM "New Session" pop-up window.

Unclosed Sessions

Session	Region	Start Date	End Date
TEST-1	: pg	: 1993/10/15:9.03	: -99.000000
vishnu3	: pgl	: 1993/10/21:11.80	: -99.000000
Dharmesh	: AG	: 1993/10/21:15.25	: -99.000000
Dharmesh	: AB	: 1993/10/21:13.05	: -99.000000
Dharmesh	: AG	: 1993/10/21:13.17	: -99.000000

Ok Edit

Figure 24. NSM "Unclosed Session" pop-up window.

5 NIDAS DATA

5.1 The NIDAS Database

The NIDAS database is completely described in Appendix C.

5.2 NIDAS User Configuration Data

User configuration data is contained in the file `nidas_config.def`. A complete description of the NIDAS configuration data can be found in Appendix H.

6 NIDAS DATA FILES

NIDAS retrieves data from the NIDAS database except for the user configuration file. There are no other data file requirements for NIDAS.

7 REQUIREMENTS TRACEABILITY

Functional requirements (NFR) and design requirements (NDR) have been defined for NIDAS. NFR/NDR requirements are also indicated in subsections 3.1, 3.2 and 3.3 for cross-reference and traceability. CSC responsibility for achieving each NFR and NDR are indicated parenthetically in the following descriptions for each NFR and NDR.

NFR1: Operate in an interactive manner; i.e., displays must be interactive. (GUI/DRM/DIM)

NFR2: Provide overlay capability for several different types of ocean, satellite and meteorological data. (GUI)

NFR3: The final product will be a three dimensional gridded field of temperature and salinity profiles. (GUI/DRM/DIM/NSM)

NFR4: System must be able to manipulate overlays of various data types (point, grid, image). (DRM)

NFR5: System must be able to graphically construct and extend temperature and salinity profiles onto a geographic grid or polygon. (DIM/NSM)

NFR6: System must be able to generate three-dimensional gridded fields from a combination of profiles and gridded data. (DIM/NSM)

NFR7: There must be two options for final grid display:

a. Interpolation. (DIM/DRM/NSM)

b. Area fill with provincial or synthetic profiles (retaining polygon concept).

(DIM/DRM/NSM)

NFR8: Access to Regional MOODS. (DRM)

NFR9: Access to Regional Bathymetry. (DRM)

NFR10: Access to High Resolution GDEM. (DRM)

NFR11: Access to Bathythermograph Observations. (DRM)

NFR12: Access to GOODS Buoys. (DRM)

NFR13: Access to Coastlines/Shorelines. (DRM)

NFR14: Access to MCSST (grids and satellite imagery) (DRM)

NFR15: Access to GOODS. (DRM)

NFR16: Access to OOC Frontal Composite. (DRM)

NFR17: Access to NAVOCEANO Regional MODAS. (DRM)

NFR18: Link to Circulation Model. (DRM)

NFR19: Link to Ice Products (NAVPOLAROCEANCEN). (DRM)

NFR20: Link to Altimetry Data. (DRM)

NFR21: Link to Acoustic Analysis Guidance Product(s). (DRM)

NFR22: Selectively evaluate and/or edit environmental data. (DRM/DIM)

NFR23: Selectively retain subsets of environmental data after evaluation and/or editing procedures have been performed. (GUI)

NFR24: Selectively replace portions of grid-formatted data with "provinced profiles".

(DIM/GUI/NSM)

NFR25: Construct profiles for Profile Composite Chart using MOODS, GOODS, CLIMO and MODAS as input. (DRM)

NFR26: Main Chart must display data distribution points (profile locations), frontal locations (lines), contoured data fields, vertical cross sections (MODAS/CLIMO contours or profiles), coastlines, 5' bathymetry contours, images (satellite data) and time specification information. (DRM/GUI)

NFR27: Main Chart must support construction and overlay of transects and polygons. (DIM/GUI)

NFR28: Profiles on Main Chart must be viewable on envelope or inside envelope. (GUI)

NFR29: Main Chart and Profile Composite Chart must have zoom capability. (DIM/GUI)

NFR30: Allow province definition on Main Chart. (DIM/GUI/NSM)

NFR31: Insert selected profile inside provinces. (DIM/GUI/NSM)

NFR32: Store province and profile in the database. (NSM)

NFR33: Display province (as overlay) at any time. (DIM/GUI/NSM)

NFR34: Top-most province masks underlying province grid points. (DIM/GUI)

NFR35: Allow display of province and profile at any time. (DIM/GUI)

NFR36: Construct SST band from a polygon (based on min and max temperature). (DIM/GUI)

NFR37: Allow toggle to control display of SST band on Profile Composite Plot. (DIM/GUI)

NFR38: Plot MODAS for different days (same depth) on Main Chart. (NSM/GUI/DIM)

NFR39: Pre-select CLIMO depths (0, 50, 100, 200, 400); store in config. file. (GUI)

NFR40: Toggle between temp and sal in Profile Composite Chart. (GUI)

NFR41: Create synthetic temp or sal profiles in Profile Composite Chart. (DIM/GUI)

NFR42: Draw corresponding salinity profiles for temperature profiles displayed on Profile Composite Chart. (GUI)

NFR43: System must be able to establish a new session. (NSM)

NFR44: Access an unclosed session. (NSM)

NFR45: Access a closed session. (NSM)

NFR46: System must be able to close a new or unclosed session. (NSM)

NFR47: System must be able to turn off a new or unclosed session. (NSM)

NIDAS DESIGN REQUIREMENTS (NDR):

NDR1: NIDAS must operate as a stand-alone system. (CSCI)

NDR2: NIDAS must operate within the UNIX operating system environment. (CSCI)

NDR3: NIDAS must execute within the X-Windows client-server model. (CSCI)

NDR4: Window displays must incorporate the Open Software Foundation (OSF) Motif Widget Library. (CSCI)

NDR5: There must be a relational database management system (rdbms) specifically for NIDAS utilization. (DRM)

NDR6: NIDAS must include an internal link to the rdbms for data retrieval. (DRM)

NDR7: Ingestion of data into the rdbms will be accomplished by software external to NIDAS. (N/A)

NDR8: NIDAS must access regional subsets of core data bases through IDBMS (when functional). (DRM)

NDR9: Zoom capability for NIDAS Main Chart must replace enlarged display within Main Chart window; i.e. no pop-up windows for enlarged area. (GUI/DIM)

NDR10: T vs. Z axes in Profile Composite Chart are to be specified in configuration file. (CSCI)

NDR11: Final product: 3-D field (T and S). (GUI/DRM/DIM)

8 NOTES

A glossary of terms is contained in Appendix A. A listing of acronyms is contained in Appendix B. Appendix C is the NIDAS RDBMS Specification. Appendix D lists NIDAS RDBMS functional (DBFR) and design (DBDR) requirements. Appendix E discusses the NIDAS

development environment. Appendix F contains modifications and changes to the NIDAS design. Appendix G is a listing of NIDAS structures and the zoom.h header file. Appendix H provides an explanation of the user default file "nidas_config.def".

APPENDIX A. GLOSSARY OF TERMS

Computer Software Configuration Item (CSCI) - a software application or a major component thereof.

Computer Software Component (CSC) - a top level functional module within a computer software configuration item (CSCI). CSC's are generally considered to be one structural level below the CSCI.

Computer Software Unit (CSU) - low level software modules, usually at the function or subroutine level that perform specific functions within a CSC.

Data Interactive Module (DIM) - NIDAS module that performs data manipulation functions and processing required for display and interpretation of data.

Data Retrieval Module (DRM) - NIDAS module responsible for identifying, obtaining and formatting data obtained from the NIDAS (NEONS) database.

Graphical User Interface (GUI) - NIDAS module responsible for interfacing with the user and controlling the functionality of the main NIDAS display.

NIDAS Session Module (NSM) - NIDAS Module responsible for establishing a session for inserting provinces and its associated polygons.

Pixmap - "... is a window like structure memory in which graphics are drawn." This graphics can be copied to the window.

Widget - "... a graphic device capable of receiving input from the keyboard and the mouse and communicating with an application or another widget by means of a callback. Every widget is a member of only one class and always has a window associated with it." (from: OSF/Motif Programmer's Guide, Rev. 1.1, Open Software Foundation, Prentice Hall, Englewood Cliffs, NJ, 1991, p. GL-13.)

APPENDIX B. LIST OF ACRONYMS

CAST - Center for Air Sea Technology
Climo - climatology
CSC - Computer Software Component
CSCI - Computer Software Configuration Item
CSU - Computer Software Configuration Unit
DBDR - Database Design Requirement
DBFR - Database Functional Requirement
DOD - Department of Defense
DIM - Data Interactive Module
DRM - Data Retrieval Module
GDEM - General Digital Environmental Model
GOODS - Global Oceanographic Observation Data Set
GUI - Graphical User Interface
MODAS - Modular Ocean Data Assimilation System
MOODS - Master Oceanographic Observation Data Set
MSU - Mississippi State University
NASA - National Aeronautics and Space Administration
NAVOCEANO - Naval Oceanographic Office
NEONS - Navy Environmental Operational Nowcast System
NDR - NIDAS Data Requirement
NFR - NIDAS Functional Requirement
NIDAS - Naval Interactive Data Analysis System
NSM - NIDAS Session Module
OTIS - Optimal Thermal Interpolation System
OSF - Open Software Foundation
PMI - Program Modernization Initiative
RDBMS - Relational Database Management System
SQL - Structured Query Language

APPENDIX C. THE NIDAS RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS) SPECIFICATION

NIDAS accesses a NEONS database created specifically to support the CSCI. The database exists as a dedicated external and independent element of the system. Data ingestion into the database is also accomplished independently of NIDAS. NIDAS queries the database and retrieves requested data from it by calling NEONS software library functions. NEONS provides a data model for all generic data types (gr'd, volume, latitude/longitude/time, line, grid, image and geographical) that are required by NIDAS. These data types support the following operational datasets:

1. bathythermograph observations in standard format (lft)
2. Modular Ocean Data Assimilation System (MODAS) datasets (volume)
3. General Digital Environmental Model (GDEM) dataset (volume)
4. Coastlines (geographical)
5. Master Oceanographic Observation Data Set (MOODS) observations (lft)
6. Global Oceanographic Observation Data Set (GOODS) observations(lft)
7. Ocean fronts and eddies (line)
8. Satellite imagery (image)
9. NIDAS (volume)

Each dataset can be retrieved by the following parameters:

MODAS - parameter, date, and levels
Climo - Parameter, season, and levels
Coastline - specify the resolution (1,3,8, or 20 km)
MOODS - lat, lon, time, month, water depth, parameter, source, instrument,
 classification, cruise id
GOODS - lat, lon, time, parameter
Fronts - dates
Eddies - dates
Image - dates
NIDAS - parameter, date, and levels

Further information about NEONS, its structure and use is available in the NEONS design document ["Database Design Document for the Naval Environmental Operational Nowcast System", Version 3.5, Naval Research Laboratory (NRL), Monterey, CA 93943-5006]. In addition, the NEONS installation package includes source code and manual pages for each major functional element. NEONS version control is managed by the Naval Research Laboratory, Monterey, CA.

APPENDIX D. FUNCTIONAL AND DESIGN REQUIREMENTS FOR THE NIDAS RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS)

DATABASE FUNCTIONAL REQUIREMENTS (DBFR):

DBFR1: Database must be capable of data retrieval.

DBFR2: Data ingestion into the rdbms is to be accomplished external to NIDAS as data becomes available using automated processes (crontab, etc.).

DBFR3: Data resident in rdbms to include the last four past versions (at a minimum) in addition to the current dataset in a revolving data pool.

DATABASE DESIGN REQUIREMENTS (DBDR):

DBDR1: RDBMS engine must be Empress.

DBDR2: RDBMS model must be the Naval Environmental Operational Nowcast System (NEONS) Version 3.5.1, or later.

DBDR3: Database must be compatible with NAVOCEANO Program Modernization Initiative (PMI) of 26 January 1993.

APPENDIX E. THE NIDAS DEVELOPMENT ENVIRONMENT

NIDAS has been developed in a Sun Microsystems SparcStation model 10 computer hardware environment. The operating system was SUNOS version 4.1.3, including the resident SUN C compiler which was used to write the NIDAS software code. Some minor elements of NEONS have been written in FORTRAN77 (Sun FORTRAN77 version 1.4). Graphics support is provided by UNIRAS ag/X Toolmaster version 6v3b. The RDBMS engine is Empress version 6.2. The windowing environment consists of X-Windows version X11 R5 and the OSF Motif widget set version 1.3.

APPENDIX F. MODIFICATIONS AND CHANGES TO THE NIDAS DESIGN

APPENDIX G. NIDAS STRUCTURES

A C structure is a collection of data variables, pointers or other structures grouped together for the convenience of using a single variable name to reference or identify the whole group. The use and behavior of structures is covered in any credible C programming language tutorial or textbook.

The **NIDAS** structure is a collection of 20 pointers to subordinate structures that comprise the critical data framework of the NIDAS application. The **NIDAS** structure is defined as follows:

```
typedef struct {
    FRONT_WINDOW    *fWindow;
    EDIT_FLAGS      *eflags;
    TIME_LOC        *locTime;
    BATHY_STRUCT     *bathyStruct;
    COAST_STRUCT     *coastStruct;
    OTIS_STRUCT      *otisStruct;
    GDEM_STRUCT      *gdemStruct;
    MOODS_STRUCT     *moodsStruct;
    GOODS_STRUCT     *goodsStruct;
    FRONT_STRUCT     *frontStruct;
    EDDY_STRUCT      *eddyStruct;
    OTISPROF_STRUCT  *otisProfStruct;
    GDEMPROF_STRUCT  *gdemProfStruct;
    IMAGE_STRUCT     *imageStruct;
    NIDASZOOMDIALOG  *zoom;
    NIDAS3D_STRUCT   *nidas3dStruct;
    COLOR_STRUCT     *colorStruct;
    VERT_XSEC_STRUCT *vert_xsec;
    EXPORT_STRUCT    *exportStruct;
    SYNTH_TOGGLE     *synthToggle;
} NIDAS;
```

The following are source code listings for each elemental data structure contained in the **NIDAS** structure:

The **DEFAULT_STRUCT** Structure

```
typedef struct {
    char  day1_color[11];
    char  day2_color[11];
    char  day3_color[11];
    char  sstColor[11];
    char  poly1Color[11];
    char  poly2Color[11];
    char  poly3Color[11];
    char  *parm[MAX_PARM];
    char  front_color[11];
    char  coast[15];
    char  coast_color[11];
    char  bathy_color[11];
}
```

```

char    xlabel[7];
char    ylabel[7];
char    minMoodsDate[15];
char    maxMoodsDate[15];
char    otisFile[61];
char    gdemFile[61];
char    moodsFile[61];
char    goodsFile[61];
char    volFile[61];
double  min_lat;
double  max_lat;
double  min_lon;
double  max_lon;
long    min_wdepth;
long    max_wdepth;
long    min_class;
long    max_class;
long    min_cruise;
long    max_cruise;
long    min_inst;
long    max_inst;
long    min_source;
long    max_source;
long    min_month;
long    max_month;
long    min_parm;
long    max_parm;
float   min_val;
float   max_val;
float   min_axis;
float   max_axis;
float   x_delta;
float   y_delta;
float   bathy_step;
float   level;
int     num_parm;
int     center_date;
int     day1;
int     day2;
int     day3;
int     moods_julian_date;
} DEFAULT_STRUCT;
DEFAULT_STRUCT *defStruct;

```

The MAIN_WINDOW Structure

```

typedef struct {
Widget colorText;
Window window;
GC      gc;
GC      rgc;

```

```

Pixmap pixmap;
Pixmap pixmap1;
Boolean ifThereIsPixmap;
Boolean isCallback;
char *xlabel;
char *ylabel;
double min_lat;
double max_lat;
double min_lon;
double max_lon;
float xmin;
float xmax;
float ymin;
float ymax;
float x_delta;
float y_delta;
float lon_arr[MAX_POINTS];
float lat_arr[MAX_POINTS];
float polyxmax;
float polyxmin;
float polyymin;
float polyymin;
float sstMinVal;
float sstMaxVal;
int width;
int height;
int depth;
int xexpose;
int yexpose;
int lon_range;
int lat_range;
int proj_type;
int xstart; /* X coordinate of rubberband origin */
int ystart; /* Y coordinate of rubberband origin */
int xlast; /* X coordinate of rubberband extreme */
int ylast;
int polyPoints;
int backupPoints;
int overlayColor;
int defOverlayColor;
} MAIN_WINDOW;

```

The PROFILE_WINDOW Structure

```

typedef struct {
Window window;
GC gc;
Pixmap pixmap;
Pixmap pixmap1;
Boolean ifThereIsPixmap;
float xarr[MAX_POINTS];

```

```

float yarr[MAX_POINTS];
float xpts[MAX_POINTS];
float ypts[MAX_POINTS];
float synxarr[50];
float synyarr[50];
float xmin;
float xmax;
float ymin;
float ymax;
float polyxmax;
float polyxmin;
float polyymin;
float polyymin;
int polyPoints;
int backupPoints;
int width;
int height;
int depth;
int xexpose;
int yexpose;
int numOfSynPoints;
} PROFILE_WINDOW;

```

The FRONT_WINDOW Structure

```

typedef struct {
MAIN_WINDOW mainWindow;
PROFILE_WINDOW profileWindow;
Widget draw1;
Widget draw2;
Display *display;
Widget remark;
Widget front_page;
Widget xlabel;
Widget xtext;
Widget ylabel;
Widget ytext;
int windowId;
} FRONT_WINDOW;

```

The EDIT_FLAGS Structure

```

typedef struct {
Boolean is_polygon;
Boolean is_province;
Boolean is_zoom;
Boolean is_transect;
Boolean is_session;
} EDIT_FLAGS;

```

The TIME_LOC Structure

```
typedef struct {  
    DATE start_date;  
    DATE end_date;  
    double min_lat;  
    double min_lon;  
    double max_lat;  
    double max_lon;  
    double start_hour;  
    double end_hour;  
} TIME_LOC;
```

The BATHY_STRUCT Structure

```
typedef struct {  
    Widget max_text;  
    Widget min_text;  
    Widget step_text;  
    Widget color_text;  
    Pixmap pixmap;  
    Boolean ifThereIsPixmap;  
    double *lon;  
    double *lat;  
    float minVal;  
    float maxVal;  
    float defMinVal;  
    float defMaxVal;  
    float *bathy;  
    float *projLon;  
    float *projLat;  
    float step;  
    float defStep;  
    int color;  
    int defColor;  
    int rowcnt;  
    int colcnt;  
} BATHY_STRUCT;
```

The COAST_STRUCT Structure

```
typedef struct {  
    Widget coast_text;  
    Widget color_text;  
    Pixmap pixmap;  
    Boolean ifThereIsPixmap;  
    char coast[15];  
    char defCoast[15];  
    int color;  
    int defColor;  
} COAST_STRUCT;
```

The OTIS_STRUCT Structure

```
typedef struct {
Widget lvl_list;
Widget parm_list;
Widget date_list;
Widget min_text;
Widget max_text;
Widget step_text;
Widget color_text;
Pixmap pixmap;
Boolean ifThereIsPixmap;
Boolean isData;
Boolean isOption;
int parmFlag;
char parm[21];
char date[15];
double *lon;
double *lat;
float *projLon;
float *projLat;
float *bathy;
float minVal;
float maxVal;
float defMinVal;
float defMaxVal;
float *data;
float step;
float defStep;
float lvl_val[40];
float level;
float x_int_dis;
float y_int_dis;
int color;
int defColor;
int lvl_cnt;
int rowcnt;
int colcnt;
} OTIS_STRUCT;
```

The NIDAS3D_STRUCT Structure

```
typedef struct {
Widget lvl_list;
Widget parm_list;
Widget date_list;
Widget min_text;
Widget max_text;
Widget step_text;
Widget color_text;
```

```

Pixmap pixmap;
Boolean ifThereIsPixmap;
Boolean isData;
Boolean isOption;
int parmFlag;
char  parm[21];
char  date[15];
double *lon;
double *lat;
float  *projLon;
float  *projLat;
float  *bathy;
float  minVal;
float  maxVal;
float  defMinVal;
float  defMaxVal;
float  *data;
float  step;
float  defStep;
float  lvl_val[40];
float  level;
float  x_int_dis;
float  y_int_dis;
int    color;
int    defColor;
int    lvl_cnt;
int    rowcnt;
int    colcnt;
} NIDAS3D_STRUCT;

```

The GDEM_STRUCT Structure

```

typedef struct {
Widget lvl_list;
Widget parm_list;
Widget month_list;
Widget min_text;
Widget max_text;
Widget step_text;
Widget color_text;
Pixmap pixmap;
Boolean ifThereIsPixmap;
Boolean isData;
Boolean isOption;
int parmFlag;
char parm[21];
char month[15];
double *lon;
double *lat;
float  *projLon;
float  *projLat;

```

```

float *bathy;
float minVal;
float maxVal;
float defMinVal;
float defMaxVal;
float *data;
float step;
float defStep;
float lvl_val[40];
float level;
float x_int_dis;
float y_int_dis;
int color;
int defColor;
int lvl_cnt;
int rowcnt;
int colcnt;
} GDEM_STRUCT;

```

The FRONT_STRUCT Structure

```

typedef struct {
Widget date_list;
Widget color_text;
Pixmap pixmap;
Boolean ifThereIsPixmap;
Boolean isData;
char date[15];
float freddy;
int color;
int defColor;
} FRONT_STRUCT;

```

The EDDY_STRUCT Structure

```

typedef struct {
Widget date_list;
Widget color_text;
Pixmap pixmap;
Boolean ifThereIsPixmap;
Boolean isData;
char date[15];
float freddy;
int color;
int defColor;
} EDDY_STRUCT;

```

The GOODS_HEADER Structure

```

typedef struct {
double minLat;

```

```

double maxLat;
double minLon;
double maxLon;
long minParm;
long maxParm;
char minDate[15];
char maxDate[15];
char defMinDate[15];
char defMaxDate[15];
double defMinLat;
double defMaxLat;
double defMinLon;
double defMaxLon;
long defMinParm;
long defMaxParm;
int parm;
int numOfProfiles;
float *lon;
float *lat;
int selection_type;
Pixmap mainPixmap;
Pixmap profilePixmap;
Boolean ifThereIsPixmap;
Boolean isData;
Boolean isSst;
Widget time_toggle;
Widget min_toggle;
Widget max_toggle;
Widget parm_scale;
Widget min_date_scale;
Widget max_date_scale;
Widget min_date_scaleText;
Widget max_date_scaleText;
Widget lat_text;
Widget lon_text;
Widget min_lat_text;
Widget max_lat_text;
Widget min_lon_text;
Widget max_lon_text;
Widget min_time_text;
Widget max_time_text;
Widget min_parm_text;
Widget max_parm_text;
int day1_color;
int day2_color;
int day3_color;
int defDay1Color;
int defDay2Color;
int defDay3Color;
Widget day1_color_text;
Widget day2_color_text;

```

```

Widget day3_color_text;
int day1;
int day2;
int day3;
int defDay1;
int defDay2;
int defDay3;
Widget day1_text;
Widget day2_text;
Widget day3_text;
int centerDate;
int sstColor;
int defCenterDate;
int defSstColor;
int poly1Color;
int poly2Color;
int poly3Color;
int defPoly1Color;
int defPoly2Color;
int defPoly3Color;
Widget poly1_color_text;
Widget poly2_color_text;
Widget poly3_color_text;
Widget sst_color_text;
Widget center_date_text;
} GOODS_HEADER;

```

The GOODS_DATA Structure

```

typedef struct {
double lat;
double lon;
double hour;
DATE date;
float *parm[3];
float *depth;
float clasId;
int numOfPoints;
int julian;
} GOODS_DATA;

```

The GOODS_STRUCT Structure

```

typedef struct {
GOODS_HEADER *goodsHeader;
GOODS_DATA *goodsData[20000];
} GOODS_STRUCT;

```

The MOODS_DATA Structure

```
typedef struct {
double lat;
double lon;
double hour;
DATE date;
float parm_1_name;
float hdr_txt[60];
float nprof;
float prof_flag[8];
float jprof;
float ipat;
float imass;
float improv;
float moods_bot_dpth;
float cyc1_cnt;
float cyc2_cnt;
float extra;
float rpln_cnt;
float clas_num;
float inst_num;
float src_num;
float cruise_num;
float *parm[3];
float *depth;
int numOfPoints;
int julian;
char ident[10];
} MOODS_DATA;
```

The MOODS_HEADER Structure

```
typedef struct {
char minDate[15];
char maxDate[15];
double minLat;
double maxLat;
double minLon;
double maxLon;
long minClass;
long maxClass;
long minInst;
long maxInst;
long minSource;
long maxSource;
long minMonth;
long maxMonth;
long minWdepth;
long maxWdepth;
long minCruise;
```

```

long  maxCruise;
long  minParm;
long  maxParm;
int   month;
int   defMonth;
int   selection_type;
int   parm;
Widget class_list;
Widget inst_list;
Widget source_list;
Widget month_list;
Widget cruise_id_text;
Widget water_depth_text;
Widget lat_text;
Widget lon_text;
Widget parm_scale;
Widget time_toggle;
Widget min_toggle;
Widget max_toggle;
Widget min_lat_text;
Widget max_lat_text;
Widget min_lon_text;
Widget max_lon_text;
Widget min_class_text;
Widget max_class_text;
Widget min_inst_text;
Widget max_inst_text;
Widget min_src_text;
Widget max_src_text;
Widget min_month_text;
Widget max_month_text;
Widget min_wdepth_text;
Widget max_wdepth_text;
Widget min_parm_text;
Widget max_parm_text;
Widget min_cruise_text;
Widget max_cruise_text;
Widget min_time_text;
Widget max_time_text;
int   numOfProfiles;
float *lon;
float *lat;
Pixmap mainPixmap;
Pixmap profilePixmap;
Boolean ifThereIsPixmap;
Boolean selectTime;
Boolean selectMonth;
Boolean isSst;
int   day1_color;
int   day2_color;
int   day3_color;

```

```

int  defDay1Color;
int  defDay2Color;
int  defDay3Color;
Widget day1_color_text;
Widget day2_color_text;
Widget day3_color_text;
int  day1;
int  day2;
int  day3;
int  defDay1;
int  defDay2;
int  defDay3;
Widget day1_text;
Widget day2_text;
Widget day3_text;
int  centerDate;
int  sstColor;
int  defCenterDate;
int  defSstColor;
int  poly1Color;
int  poly2Color;
int  poly3Color;
int  defPoly1Color;
int  defPoly2Color;
int  defPoly3Color;
Widget poly1_color_text;
Widget poly2_color_text;
Widget poly3_color_text;
Widget sst_color_text;
Widget center_date_text;
} MOODS_HEADER;

```

The DEF_M_HEADER Structure

```

typedef struct {
char  defMinDate[15];
char  defMaxDate[15];
double defMinLat;
double defMaxLat;
double defMinLon;
double defMaxLon;
long  defMinClass;
long  defMaxClass;
long  defMinInst;
long  defMaxInst;
long  defMinSource;
long  defMaxSource;
long  defMinWdepth;
long  defMaxWdepth;
long  defMinMonth;
long  defMaxMonth;

```

```

long defMinParm;
long defMaxParm;
long defMinCruise;
long defMaxCruise;
int defMonth;
} DEF_M_HEADER;

```

The MOODS_STRUCT Structure

```

typedef struct {
MOODS_HEADER *moodsHeader;
DEF_M_HEADER *defMoodsHeader;
MOODS_DATA *moodsData[20000];
} MOODS_STRUCT;

```

The OTISPROF_STRUCT Structure

```

typedef struct {
Boolean isButtonOn;
int color;
int defColor;
Widget color_text;
float data[21];
} OTISPROF_STRUCT;

```

The GDEMPROF_STRUCT Structure

```

typedef struct {
Boolean isButtonOn;
int color;
int defColor;
Widget color_text;
float data[21];
} GDEMPROF_STRUCT;

```

The IMAGE_STRUCT Structure

```

typedef struct {
Boolean ifThereIsPixmap;
Boolean isData;
Pixmap pixmap;
long imageId;
float minVal;
float maxVal;
float step;
float defMinVal;
float defMaxVal;
float defStep;
float minTemp;
float maxTemp;

```

```

Widget image_list;
Widget min_text;
Widget max_text;
Widget step_text;
REG_GEOM geom;
unsigned short *sbuff;
unsigned short *newImage;
int rowcnt;
int colcnt;
int minPixel;
int maxPixel;
double min_lon;
double max_lon;
double min_lat;
double max_lat;
} IMAGE_STRUCT;

```

The SESSION Structure

```

typedef struct{
char region[21];
char session[21];
char type;
char mode;
char classification[15];
char userName[15];
long uid;
long sessionId;
float maxDepth;
double min_etm;
double max_etm;
} SESSION;

```

The SYNTHETIC Structure

```

typedef struct {
float xvalues[MAX_POINTS];
float yvalues[MAX_POINTS];
float temp[50];
float salinity[50];
float depth[50];
int prvncePoints;
int synthPoints;
double id;
char date[15];
char prvnceOrigin;
char synthOrigin;
} SYNTHETIC;

```

The SYNTH_STRUCT Structure

```
typedef struct {  
    SESSION    session;  
    SYNTHETIC  synthetic[20];  
} SYNTH_STRUCT;
```

Initialized struct SYNTH_STRUCT Variables

```
SYNTH_STRUCT synthStruct1;  
SYNTH_STRUCT synthStruct2;  
SYNTH_STRUCT synthStruct3;  
SYNTH_STRUCT *synth;
```

The SYNTH_TOGGLE Structure

```
typedef struct {  
    Widget new;  
    Widget unclosed;  
    Widget repeat;  
    Widget display;  
    Widget data;  
    Widget options;  
    Widget close;  
    Widget turn_off;  
} SYNTH_TOGGLE;
```

The COLOR_STRUCT Structure

```
typedef struct {  
    unsigned long red;  
    unsigned long blue;  
    unsigned long white;  
    unsigned long cyan;  
    unsigned long cyan4;  
    unsigned long tan;  
    unsigned long whiteyellow;  
    unsigned long skyblue;  
    unsigned long color9;  
    unsigned long color10;  
} COLOR_STRUCT;
```

The VERT_XSEC_STRUCT Structure

```
typedef struct {  
    float pt_lat[2], pt_lon[2];  
    int  num_poly_pts;  
    float gtlat[200], gtlon[200], gtd[200];  
    float *data;
```

```

int iend;
Widget apply;
char parm_name[31];
float min_val, max_val;
int int_val;
int dataType;
int lvl_cnt;
int rowcnt;
int colcnt;
float spacing;
float xmax;
float *depth;
float minDepth;
float maxDepth;
float interval;
float offset;
Widget transectDialog;
} VERT_XSEC_STRUCT;

```

The EXPORT_STRUCT Structure

```

typedef struct {
Widget header_text;
Widget otis_text;
Widget gdem_text;
Widget moods_text;
Widget goods_text;
Widget vol_text;
char header[60];
char otisFile[51];
char gdemFile[51];
char moodsFile[51];
char goodsFile[51];
char volFile[51];
int fileType;
} EXPORT_STRUCT;

```

LISTING OF THE "Zoom.h" HEADER FILE

```

/* NAME Zoom */
/* PACK h */
/* DEVT I */
/* VERS 910509 */
/* HIST 6V2 */

/* EXTRACT=Zoom_h */
/*-----*
*
* Zoom.h - Header file for CAST/Motif example programs which use *

```

```

*      the Zoom dialog      *
*                               *
*-----*/

/* Prevent the contents of this file from being included more than once */

#ifndef _Zoom_h
#define _Zoom_h

/*-----*/

typedef struct _NIDASZOOMDIALOG {
Widget      dialog; /* Pointer to associated DialogShell */
Widget      drawing; /* Pointer to associated DrawingArea */
Display*     display; /* Pointer to display */
Window       window; /* Pointer to graphics window */
unsigned int width; /* Width of graphics window */
unsigned int height; /* Height of graphics window */
unsigned int depth; /* Depth of graphics window */
Pixmap       pixmap; /* Backing pixmap for graphics window */
GC           gc; /* GC for Xlib operations */
int          backing; /* Indicates server backing store */
int          plotted; /* Indicates if graphics are visible */
int          startx; /* X grid coordinate of first zoom corner */
int          starty; /* Y grid coordinate of first zoom corner */
int          endx; /* X grid coordinate of second zoom corner */
int          endy; /* Y grid coordinate of second zoom corner */
float        xmin;
float        xmax;
float        ymin;
float        ymax;
int          xgrid; /* Width of subgrid */
int          ygrid; /* Height of subgrid */
Boolean      selected;
Boolean      zoomOnly;
Boolean      overlay;
} NIDASZOOMDIALOG;

/*-----*/

#endif /* _Zoom_h */
/* DON'T ADD ANYTHING AFTER THIS #endif */
/* ENDEXTRACT */

```

APPENDIX H. THE NIDAS DEFAULT CONFIGURATION FILE

Default values for parameters that are critical to NIDAS operation are maintained in a user default file. Users should not create their own user default file. Instead, the file should be provided by NIDAS site managers. Site managers should ensure that file privileges are restricted to "read only". Any changes to this file could cause the application to fail. All values and colors may be changed through interaction with NIDAS, except minlat, maxlat, minlon, and maxlon. When launching NIDAS, the application will look for the file "nidas_config.def" in the current directory. If the "nidas_config.def" file cannot be located, the application will fail. The format for the "nidas_config.def" default file is as follows:

```
22.0 31.0 47.0 74.0 /* min and max latitude and longitude */
3.0 1.0 /* longitude and latitude axis ticks */
wvs_8km_cst /* resolution of coastline */
WHITE /* color of the coastline 8/
ORANGE 100.0 /* Bathymetry color and contour interval */
10.0 35.0 0.0 400.0 /* min and max temperature and depth */
Temp /* x axis label */
Depth /* y axis label */
3 /* number of parameters */
temp /* temperature */
sal /* salinity */
sndspd /* sound speed */
YELLOW /* color of MODAS contour */
center_date 30 /* Center date for Moods data */
GREEN 15 /* Day 1 time color and window */
GREEN 0 /* Day 2 time color and window */
GREEN 0 /* Day 3 time color and window */
sst_color BLUE /* Sea surface temperature color */
polygon1_color ORANGE /* day 1 polygon color */
polygon2_color YELLOW /* day 2 polygon color */
polygon3_color WHITE /* day 3 polygon color */
classification 10001000 10001000 /* min and max classification for Moods */
source_code 2 31 /* min and max source code for Moods */
inst_type 0 33 /* min and max instrument type for Moods */
months 1 2 /* min and max months for Moods */
cruise 0 151375 /* min and max cruise number for Moods */
parm 3 3 /* min and max parameters for Moods */
water_depth 0 3958 /* min and max water depth for Moods */
level 0.0 /* MODAS level to be plotted */
/epoch/cast/dharmesh/cast/backup/src/otis_data.out /* MODAS output file */
/epoch/cast/dharmesh/cast/backup/src/gdem_data.out /* Climo output file */
/epoch/cast/dharmesh/cast/backup/src/moods_data.out /* Moods output file */
/epoch/cast/dharmesh/cast/backup/src/goods_data.out /* Goods output file */
/epoch/cast/dharmesh/cast/backup/src/vol_data.out /* NIDAS output file */
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. Agency Use Only (Leave blank).		2. Report Date. 29 October 1993	3. Report Type and Dates Covered. TECHNICAL NOTE
4. Title and Subtitle. DESIGN DOCUMENT AND DATABASE SPECIFICATIONS FOR THE NAVAL INTERACTIVE DATA ANALYSIS SYSTEM (NIDAS) VERSION 1.0			5. Funding Numbers. Program Element No. Project No. Task No. Accession No.
6. Author(s). DHARMESH KRISHNAMAGARU, et al			
7. Performing Organization Name(s) and Address(es). MISSISSIPPI STATE UNIVERSITY CENTER FOR AIR SEA TECHNOLOGY (CAST) STENNIS SPACE CENTER MS 39529-6000			8. Performing Organization Report Number. CAST TECHNICAL NOTE 01-94
9. Sponsoring/Monitoring Agency Name(s) and Address(es). NAVAL OCEANOGRAPHIC OFFICE CODE DOST STENNIS SPACE CENTER, MS 39529			10. Sponsoring/Monitoring Agency Report Number. CAST TECHNICAL NOTE 01-94
11. Supplementary Notes. DEVELOPMENT PERFORMED UNDER NATIONAL AERONAUTICS AND SPACE ADMINISTRATION CONTRACT NUMBER NAS13-564 D.O.11			
12a. Distribution/Availability Statement. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED			12b. Distribution Code.
13. Abstract (Maximum 200 words). THE NAVAL INTERACTIVE DATA ANALYSIS SYSTEM (NIDAS) VERSION 1.0 HAS BEEN DEVELOPED TO SUPPORT OPERATIONAL OCEANOGRAPHIC ANALYSIS REQUIREMENTS OF THE NAVAL OCEANOGRAPHY COMMAND. NIDAS PROVIDES INTERACTIVE AND OVERLAY CAPABILITY FOR SEVERAL DIFFERENT TYPES OF OCEAN, SATELLITE, AND METEOROLOGICAL DATA. THE FINAL PRODUCT OF THE SYSTEM IS A THREE-DIMENSIONAL GRIDDED FIELD OF TEMPERATURE AND SALINITY PROFILES. THIS DOCUMENT DESCRIBES THE DESIGN AND SPECIFICATIONS FOR NIDAS AND ITS ASSOCIATED RELATIONAL DATABASE MANAGEMENT SYSTEM.			
14. Subject Terms. (U) DESIGN DOCUMENT (U) NIDAS (U) SOFTWARE (U) CAST			15. Number of Pages. 29
			16. Price Code.
17. Security Classification of Report. UNCLASSIFIED	18. Security Classification of This Page. UNCLASSIFIED	19. Security Classification of Abstract. UNCLASSIFIED	20. Limitation of Abstract.

Distribution List

1. Oceanographer of the Navy
U.S. Naval Observatory
34th & Massachusetts Avenue
Washington, DC 20392
2. Commanding Officer, Code DOST
Naval Oceanographic Office
Stennis Space Center, MS 39529
(3 copies)
3. Technical Director
Code OOT
COMNAVOCEANCOM
Stennis Space Center, MS 39529
4. Defense Technical Information Center
Building 5, Cameron Station
Alexandria, VA 22304-6145
(2 copies)
5. Director, Centers and Institutes
Mississippi State University
Mississippi State, MS 39762
6. Technical Director
Naval Research Laboratory
Stennis Space Center, MS 39529